1985-03

Initial design of a relational database system for NPS computer asset identification and valuation for Risk Assessment.

Thompson, Fred W.

NAVAL POSTGRADUATE SCHOOL
Monterey, California

THESIS

INITIAL DESIGN OF A RELATIONAL DATABASE
SYSTEM FOR NPS COMPUTER ASSET IDENTIFICATION
AND VALUATION FOR RISK ASSESSMENT

by

Fred W. Thompson

March 1985

Thesis Advisor: Norman R. Lyons

Approved for public release; distribution is unlimited

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) Initial Design of a Relational Database System for NPS Computer Asset Identification and Valuation for Risk Assessment		5. TYPE OF REPORT & PERIOD COVERED Master's Thesis March 1985
		6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) Fred W. Thompson		8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS Naval Postgraduate School Monterey, CA 93943		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS Naval Postgraduate School Monterey, CA 93493		12. REPORT DATE March 1985
		13. NUMBER OF PAGES 123
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office)		15. SECURITY CLASS. (of this report) UNCLASSIFIED
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution is unlimited

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

relational database design, computer risk assessment

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

Relational database technology is examined as a means to cupport computer Risk Assessment at the Naval Postgraduate School. The current methodology for conducting computer Risk Assessment within the Department of the Navy is used to construct an initial design for a relational database system. The initial design focuses on computer

(Continued)

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE

S/N 0102-LF-014-6601

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

ABSTRACT (Continued)

asset identification and valuation as the first step of the computer Risk Assessment process.

S N 0102- LF- 014- 6601

Approved for public release; distribution unlimited.

Initial Design of Relational Database System for NPS Computer Asset Identification and Valuation for Risk Assessment

by

Fred W. Thompson Jr.
Lieutenant Commander. United States Navy
B.A.. Vanderbilt University. 1974

Submitted in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE IN INFORMATION SYSTEMS

from the

NAVAL POSTGRADUATE SCHOOL
March 1985

ABSTRACT

Relational database technology is examined as a means to support computer Risk Assessment at the Naval Postgraduate School. The current methodology for conducting computer Risk Assessment within the Department of the Navy is used to construct an initial design for a relational database system. The initial design focuses on computer asset identification and valuation as the first step of the computer Risk Assessment process.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

I. INTRODUCTION

A. BACKGROUND

The rapid growth in computer technology has brought about a tremendous increase in computer applications by the federal government. Computer proliferation has been so dramatic that it has outstripped the ability to control it. In many cases, government has not even been able to maintain accurate inventories of how much computer equipment has been bought or who has it.

In 1976, "The General Accounting Office (GAO) was able only to bracket Federal data processing spending as between $3 billion and $10 billion annually. More recently, the Office of Management and Budget (OMB) has cited a figure of $5.5 billion, and the General Services Administration (GSA) has estimated the cost of software development and maintenance alone at $2.2 billion." [Ref. 1]

More serious than the problem of controlling computer acquisition, is the problem of computer security. Hand in hand with computer growth is the increased dependence on computers for everything from Early Warning Detection Systems for national defense to word processing systems for routine administration support. Increased dependence on Automatic Data Processing (ADP) to support mission accomplishment increases the need to protect those ADP resources.

ADP security is not limited to protection against wrongful disclosures or physical assault against an ADP activity. In general. ADP security encompasses threats to ADP property and capital equipment and the physical hazards to continuing operations. For example. hardware failures. failure of supporting utilities. disasters. nonavailability of personnel. neighboring hazards. tampering with input. programs. or data files used for fraudulent purposes. and interception of acoustical or electromagnetic emanations from ADP hardware are all part of the ADP security problem. ADP resources are vulnerable to a wide assortment of threats.

A problem of this magnitude requires the utmost attention and concentrated effort of the entire ADP organization. The security solution starts with the establishment. implementation. and maintenance of a physical security program.

Analysis of risks to ADP security is an essential part of a physical security program. Risk analysis involves the identification of what is at risk and what those risks are. "The primary purpose of risk analysis is to understand the security problem by identifying security risks. determining their magnitude. and identifying areas where safeguards or controls are needed." [Ref. 2] Besides determining how much protection is required. risk analysis determines how much protection already exists. Risk analysis can also be used to determine the amount of resources needed for

security and where to allocate those resources. "The aim of risk analysis is to help ADP management strike an economic balance between the impact of risks and the cost of protective measures." [Ref. 3]

The results of risk analysis provide management with information which it can use to make decisions regarding which course of action to pursue in providing security. As a result. it is best to present the estimates of loss or damage in quantitative terms.

The risk analysis process is not something that is done one time and filed away. "It must be performed periodically to stay abreast of changes in mission. facilities. and equipment." [Ref. 4] Naval activities with ADP equipment are required by OPNAVINST 5239.1A. to update their risk assessments at least every five years besides performing new assessments for any changes in equipment. software. operating procedures. or any change that might affect overall security of the system. [Ref. 5]

The federal government must approach both these security and control problems in an economical manner. The government has issued guidelines on risk analysis including an approved methodology for implementing a risk management program. The environment for computer resource control is different than that of computer security. Congress has spearheaded the effort to control the ballooning growth in computer resources by passing laws to strictly control resource acquisition.

The Brooks Act. passed in 1964. has indirectly compelled federal agencies to maintain accurate inventories of computer assets in order to defend justification for new acquisitions.

Most organizations. however. are on their own when it comes to providing means to control computer resources. Congress requires that effective control be established. but. it does not provide guidance on how to do it. In contrast. risk assessment is supported by a well defined methodology that is recommended to solve this problem.

B. THE NAVY'S NEED TO CONTROL COMPUTER ACQUISITION AND SECURITY

The Navy has made a large investment in computer resources. The automatic data processing (ADP) budget for the Navy in FY85 was approximatly $1.961.000.000.00 [Ref. 6]. This figure represents an increase over FY84 of close to 25 percent and the rate of growth has been increasing in recent years.

Management and control of these resources has become a serious problem. The ability to effectively and efficiently manage these resources will impact future ADPE acquisition in the Navy. Various committees and individual members of Congress have expressed concern about the increasing level of Federal expenditures for data processing and have requested agencies to provide information to the Congress on the

purposes for which computers are used. how they are used and the level of expenditures requested for these activities. [Ref. 7].

A significant obstacle to future automation is the demanding Federal ADPE acquisition procedure. The ADPE acquisition procedure requires extensive planning and justification documentation. Careful planning is necessary in part due to the long lead times required in the ADPE acquisition procedure. Justification of the need for ADPE is required as a result of limited resources. The idea is to achieve optimum efficiency in distribution of limited funds.

The Navy can strengthen its argument for more ADPE if it can demonstrate effective control of existing computer assets. A system that provides an accurate accounting of computer resources and utilization would also aid the Navy in making sure that its resources are optimally utilized.

Effective control of computer assets is also essential to effective computer security. Computer security is implemented through an ADP security program and an essential part of any security program is risk assessment. Risk is determined from the evaluation of threats and vulnerabilities in relationship to the assets of the ADP facility [Ref. 8]. One necessary steps in conducting risk assessment is asset identification and valuation.

C. ADP SECURITY DIRECTIVES

ADP security is receiving attention at every level of the executive branch. The Office of Management and Budget (OMB) was assigned responsibility for the oversight and policy-making functions applicable to computer systems development and acquisition by the Brooks Act of 1965. In 1972. "OMB urged private industry -- hardware manufacturers. software houses and related service industries -- to make greater capital investments in computer security. At the time. the Federal Government was concerned that its inability to protect data in computer systems -- except at very great expense -- was limiting its ability to realize the benefits of technology." [Ref. 9]

Congress. through the Brooks Act. also assigned other Federal agencies responsiblities for ADP management. The National Bureau of Standards (NBS) was directed to provide overall coordination and technical guidance to the government's efforts in developing guidelines and standards. [Ref. 10] In June 1974. NBS published Federal Information Processing Standards Publication (FIPS PUB) 31. This publication. although not a directive. was the first comprehensive study for government agencies concerning ADP physical security and risk management.

It was OMB Circular A-71 that made Federal agencies start to think seriously about computer security. The Office of Management and Budget. relaeased OMB Circular A-71 in 1978

demonstrating the concern for computer security at the highest levels of government. In Circular A-71. OMB established minimum controls for computer security and required that every agency implement them through a computer security program. [Ref. 11] In December. 1978. the Department of Defense issued DOD Directive 5200.28 entitled "Security Requirements for Automatic Data Processing (ADP) Systems". This directive established policy for the protection of classified material in ADP systems. The Department of the Navy issued OPNAVINST 5239.1 in April. 1979. This instruction implemented the requirements put forth in OMB Circular A-71 and DOD Directive 5200.28. Specifically. it directed all activities operating computer systems to appoint an ADP Security Officer who would be responsible for conducting risk assessments on a periodic basis.

In August 1982. OPNAVINST 5239.1A was released which is the current directive governing ADP security requirements. This instruction provides a comphrehensive review of the Navy's ADP security program including the Department of the Navy (DON) approved risk assessment methodology. It is from this instruction that the data requirements for the computer asset database model for this thesis are determined.

What follows is a discussion of the origins for database systems and a look at database systems technology advantages and disadvantages. This discussion supports my decision to

propose a database design for application to the risk analysis and computer control problems.

D. WHY DATABASE

A database is a collection of integrated files. It is integrated in the sense that it stores relationships among the records in those files. A database also contains a description of its own structure. The database is accessed by a usually large complex program called the Database Management System (DBMS). The DBMS fills the role of data manager. which brings together an organization's data so that it can be readily available to many different users and applications. The DBMS. besides storing data. stores a description of the format of the data.

There are two major factors that led to the development of database systems. The first was motivated by the gradual acquisition. by organizations. of large portfolios of application programs that were developed independent of one another. Coupled with the growth of corporate applications portfolios was the growth in user sophistication and the resultant demand for more information. However. most of these applications could not be used for purposes beyond that which they were specifically developed. This was due to the fact that specific applications were created without regard to other applications outside the scope of the problem each was being designed to solve. Each application was unique and

was generally not compatible with other programs. In other words, there was great difficulty in integrating application programs to work together if they were not originally designed to do so. Additionally, maintenance of the programs was becoming unmanageable due the constant demand for minor changes in the programs.

The second major factor was the development of a specific software package designed by IBM to support the APOLLO project. North American Rockwell, the prime contractor for the project, needed a way to organize and coordinate the vast collection of research and production groups and to ensure, on a technical level, that each of millions of individual components would correctly fit into the entire assembly and work properly with all the other parts supplied by all the other firms [Ref. 12].

The result was the Generalized Update Access Method (GUAM), developed in 1964 by IBM. GUAM was designed to deal with data whose logical structure was hierarchical and for second-generation hardware which relied heavily on the use of magnetic tape.

1. Advantages

Database processing provides several advantages over traditional file processing. In file processing systems, the data is stored in files which are considered to be independent of one another. Consequently, information that could be produced by processing data stored in two

different files is not available due to the artificial partitioning in a file processing system. A DBMS. in contrast. does not partition the data. Therefore more information can be produced from a given amount of data and more knowledge is available for decision making.

Another advantage of database processing is the reduction or elimination of data duplication. By reducing data redundancy. the database structure makes more efficient use of computer storage space. Additionally. the problem of data integrity is reduced. It is much easier to maintain a single instance of a data element than it is to maintain two or three instances of the same data element. Reducing data duplication can also result in faster processing and data entry.

Another advantage of database processing is that it allows for program - data independence. Programs do not have to be changed when the data structure is changed because programs access data indirectly through the DBMS. In general. only the database structure. specific programs. and sometimes the DBMS need to be changed when a field is added to a database record or a new technique for processing files is implemented. [Ref. 13]

The benefits of economies of scale are another advantage of database processing. Improvements made to the database benefit all users and not only users of a particular application.

Finally. the DBMS environment enables the user to perform more sophisticated information retrievals.

Before going on to the disadvantages of database systems. detail. Data independence proposes to alleviate the problems created by making changes in programs and files that necessitate reorganizing files or impact all other files or programs that are used in any organized system. It is also intended to solve the problems of users not being able to tailor files to a specific application need. Data independence allows any user or group of users to have a view of the database that is different from the views used by others. Thus. a change made to accommodate one user can be kept invisible to others; a change made to the structure of a file. or to the storage device characteristics. can be hidden from all users; and each user can be given a view of the database that includes only the relevant contents in a form well suited to the users needs. [Ref. 14]

Data independence is not an automatic benefit of using a database. On the contrary. adopting the database approach. with its emphasis on sharing of data. causes the problem that data independence is intended to alleviate.

2. Disadvantages

The benefits of database technology do not come without certain costs; both financial and nonfinancial. The most visible financial cost of the database approach is associated with the acquisition of a DBMS. This large and

complex software package can cost hundreds of thousands of dollars. Additionally, where existing applications protfolios are large, conversion costs associated with going to the database system approach can be expensive. Less quantifiable costs of the database approach include the possibility of reduced system reliability, emphasis on global optimization of information usage resulting in users relinquishing their power to make information decisions on a strictly local basis, and the reduction of data processing efficiency.

Database technology is troubled by the problem of efficiency. Database systems technology involves the requirement for extensive overhead processing of each file on each access to the database. If traditional throughput rates are to be maintained, additional equipment in the form of faster processors and additional memory will have to be acquired. Also, expertise is required to design, manage, support, and maintain the database.

Another problem with database systems is that they are more vulnerable to errors. In the traditional multi-file approach, any errors that were introduced into the independent and often redundant files were not likely to propagate beyond the immediate vicinity. With the database system, there is the possibility that errors could spread further and in unanticipated directions. Table I summarizes the advantages and disadvantages associated with database processing.

These advantages and disadvantages characterize database systems in general. Microcomputer systems, however, do not exhibit many of the disadvantages noted above while

TABLE I

Advantages of Database Processing

1. More Information from the Same Amount of Data
2. New Requests and One-of-a-Kind Requests More Easily Implemented
3. Elimination of Data Duplication
4. Program/Data Independence
5. Better Data Management
6. Affordable Sophisticated Programming
7. Representation of Record Relationships

Disadvantages of Data Processing

1. Expensive to develope
2. Complex
3. Recovery More Difficult
4. Increased Vulnerability to Failure

maintaining most of the advantages. This is an important argument in favor of microcomputer DBMSs that will be explored in a later chapter.

This first chapter has discussed the importance of and requirements for maintaining effective control of computer resources. The concept of database systems as a means to effect control has also been introduced. Database technology will be discussed in greater detail in later chapters. Specifically, a relational database initial design is proposed as a means to provide necessary support for a control and risk analysis program for the Naval Postgraduate School.

Chapter Two will present an overview of the relational database design process. Advantages of using the relational data model will also be addressed. Chapters Three, Four, and Five detail the specification, logical database design, and the physical database design phases respectively, to create a database to control computer resources. Chapter Six examines the process of database implementation, involving the selection of a DBMS. Chapter Seven evaluates the database design process and discusses the concept of design iterations as a means toward design optimization. Chapter Eight presents the initial design in the context of the overall database project and recommends follow-on thesis work to complete the project. Conclusions are then drawn about initial database design, the iterative design process, and database implementation. Relational database design and database administration, in the context of risk assessment, are also highlighted.

II. RELATIONAL DATABASE DESIGN OVERVIEW

This chapter concentrates on the first three phases of the system development process; user requirements analysis. logical design. and physical design. These later two phases. the logical design and physical design are part of the database design process. A short explanation of the system development process and the system design process follows. A brief definition of user is also provided.

In computer science literature. there have been many different terms used to describe the various phases of the system development process. To help simplify the discussion in this thesis. the five basic development phases will be called user requirements analysis. logical design. physical design, implementation. and maintenance. These five phases are illustrated in Figure 2.1. The products of the different phases are also shown. To differentiate between products and development phases. rectangles are used to denote phase end-products and ovals are used to denote phases. For example. the rectangle containing the "Logical Schema" is the end-product of the oval containing the "Logical Design" phase.

There have also been many terms used to describe the different phases of system design. For our purposes. the terms used in Figure 2.1 to discuss system development. will be used to discuss the design process as well.

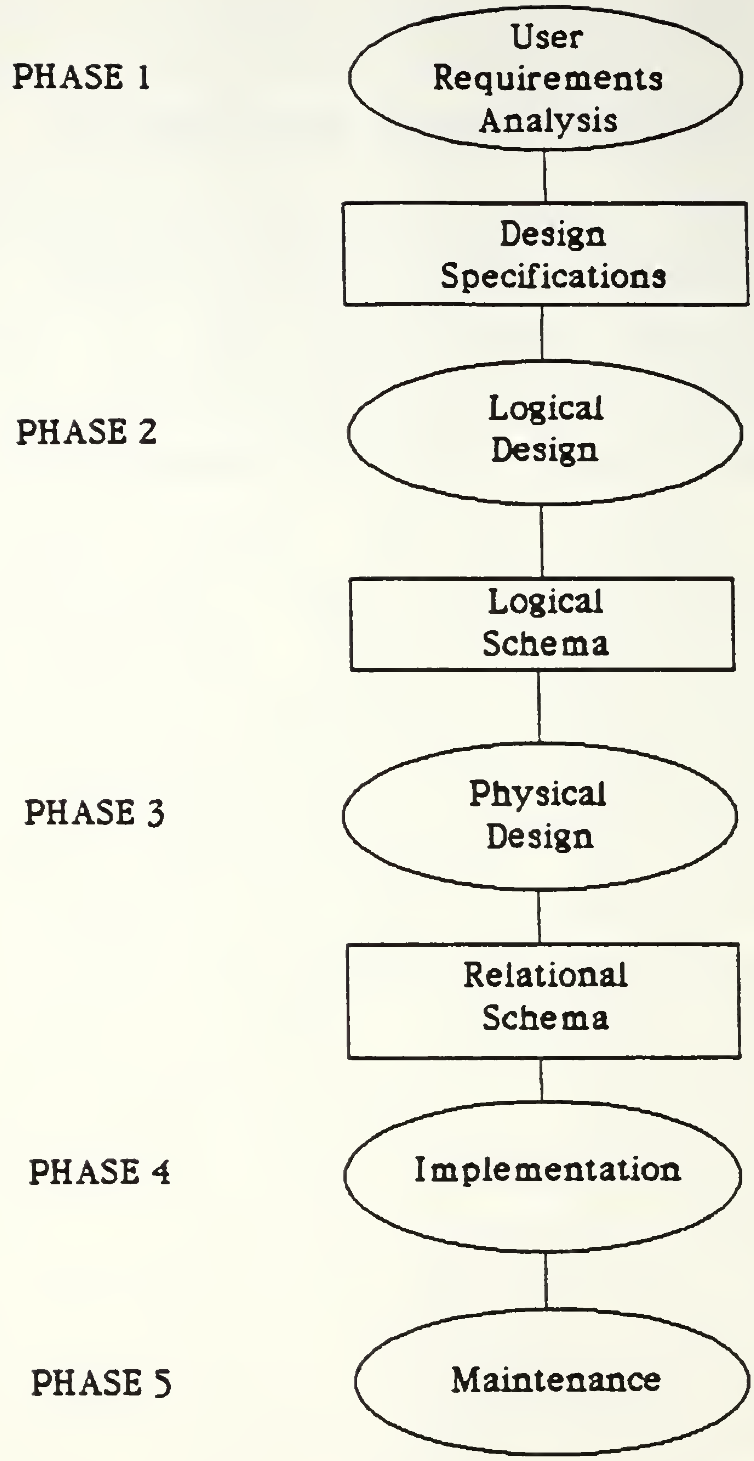


Figure 2.1 Relational Database System Development Process

The terms, system development and system design, refer to different levels of computer system development. System design is a subset of the system development process. The system design process involves two phases that fall between the user requirements analysis phase and the implementation phase of the overall system development process.

The term user, for this problem, refers to the end-user of the computer system. The computer system is designed to provide direct support to the user. Direct support is stipulated because the end-user, in this case, implies some interaction on the part of the user with the computer system.

A. THE DATABASE DESIGN PROCESS

There are two basic approaches to database design: top-down and bottom-up. Making a decision between these two philosophies is usually based on whether there is already a large investment in conventional files and programs. If this is the case, then the bottom-up approach is generally more feasible. Here, one application at a time is selected for conversion and the database is extended by stages. One problem with this approach is the final database could end up being far from optimal.

The top-down approach is the preferred approach and is best used when creating a new system, preferably where little conversion is required. The goal is to produce a rational, integrated solution to business information needs.

The top-down database design methodology follows a sequence of steps. Each step considers different problems of database design at different levels of detail. This staged approach. in contrast to ad hoc design. both formalizes the design process and simplifies it. The sequence of design stages and the design tools used at each stage is called a design methodology. [Ref. 15]

There are several different design methodologies for top-down design in use today. One class of these methodologies, the databased methodology, concentrates on enterprise data. Enterprise data refers to the facts and information that are used by, and part of the organization that is being looked at. The design begins by defining data elements and structures in the user system. Then. user functions are defined. The relationship between data and function is usually expressed graphically. Data in databased methodologies are usually described by a data model or semantic model.

It is important that a model be chosen that captures the essential features of the organization's data. The data model is an important design tool in the database development process. The Relational Model will be used to design the Computer Risk Assessment Asset Identification Database. The advantages of the Relational Model will be discussed later.

In general terms, the design process consists of two phases. The first phase concentrates on the user's

requirements and building a conceptual database structure that is a model of the organization. This is the logical database design phase. The second phase is the physical database design phase. The designer must construct the physical database given the logical design and an implementation model.

B. WHAT IS A RELATIONAL DATABASE

There are three objectives which a formal model should meet. These are:

1. It can be used to identify user requirements and present them in a way that is easily understood both by users and by computer professionals.
2. It can be easily converted to a technical implementation.
3. It provides rules and criteria for efficient logical data representation. [Ref. 16]

The relational model meets the first objective because it provides an interface that both users and designers can easily and unambiguously comprehend. This is accomplished by using tables. The organizational data is specified as a set of tables or relations.

The next objective is also satisfied with a relational model because it can be directly implemented on a machine through a DBMS. Several different systems are currently available on the market. Another solution would be to convert the relational model to a different physical structure.

The last objective relates to criteria for good logical design:

1. Each fact should be stored once in the database.
2. The database should be consistent following database operations.
3. The database should be resilient to change.

These criteria are realized through a process called normalization. "Normalization rules". as defined by William Dent, "are designed to prevent update anomalies and data inconsistencies." [Ref. 17]

Another aspect of the relational model is that it must provide languages to access relations. The relational model language is particularly successful for two reasons. First. relational languages are particularly sensitive to human factors. i.e.. provide a natural interface. Second. it has strong selective power. That is, it can retreive data that satisfy any condition covering any number of relations. [Ref. 18]

The goal of relational design is to structure the database as a set of normal relations. The process begins by defining data elements or attributes. After identifying all the data elements. determine how each element relates to the other and then assemble one-to-one related elements into records. The other half of the picture is the relational data access language which is used to both maintain the data and to turn it into information.

Figure 2.2 provides an illustration of a relation. This example of a relation, named INVENTORY, appears in nearly the same form as a paper inventory form it represents. The INVENTORY relation contains information about supply parts; it stores each PART_NUMBER, QUANTITY, DESCRIPTION, PRICE, and TOTAL.

RELATION: INVENTORY

PART_NUMBER	QUANTITY	DESCRIPTION	PRICE	TOTAL
111111	1	FAN BELT	2.00	2.00
222222	2	WHEEL	20.00	40.00
333333	1	CYLINDER	30.00	30.00
444444	3	TIRE	30.00	90.00

Figure 2.2 Relation INVENTORY

Each relation possesses the following properties:

1. There is one column in the relation for each attribute of the relation. Each such column is given a name that is unique in the relation.

2. The entries in the column come from the same domain.

3. The order of the columns or attributes in the relation has no significance.

4. The order of the rows is not significant.

5. There are no duplicate rows. [Ref. 19]

C. THE ENTERPRISE MODEL

The product of the user requirements analysis phase. referring to Figure 2.1. is called the design specification. In performing the requirements analysis, it is helpful to decompose the problem into a group of smaller modules. A major part of this analysis process focuses on the enterprise and the result of this anaylsis produces what is called the enterprise model.

The enterprise model is a high level. static abstraction of the organization. It shows the major functions performed and the flows of information in support of them.

The goal is for the database to model the enterprise it is designed to serve and be consistent with the dynamics of the organization. Events that occur in the enterprise should be represented by transactions in the model.

The model should represent all functions and data elements that are part of the enterprise. The database exists so that users can store and retrieve information about things in the real world. Naturally. it is impractical to store every minute detail about an organization. Consequently, a certain amount of data aggregation and

generalization is required by the database designer to construct a working model. Additionally, the data relationships of the organization should be represented in the enterprise model.

D. LOGICAL DATABASE DESIGN

The logical design phase is centered around the user characteristics of the database. The inputs to this phase are the system requirements and the project plan. The requirements are expressed as data flow diagrams, policy statements, and the data dictionary.

The logical design specifies the logical format of the database including the records to be maintained, their contents, and relationships among the records. [Ref. 20] The product of this phase is called the logical schema.

Logical database records are specified from analysis of the enterprise model. The number of records required is directly proportional to the level of detail of the enterprise model and consequently the relational model. The contents of the records, names of fields and their format, are also determined during the logical design phase.

The major steps in the logical design development phase are listed in Table II.

In general, the contents of the data dictionary are used to define the logical and user views. The policy statements aid the development of the descriptions of logical database

processing and the requirements are used to verify the completeness of the logical design. [Ref. 21]

TABLE II

Stages of Logical Database Design

1. Identify data to be stored
2. Consolidate and clarify data names
3. Develop the logical schema
4. Define processing
5. Review design

E. PHYSICAL DATABASE DESIGN

The second phase of database design is highly dependent on the DBMS being used. In this phase, the logical schema is transformed into the data constructs available with the particular DBMS. The outputs from this phase are the specification of the physical schema and the definition of user views.

The relational model provides a vocabulary for describing the structure and processing of the database. These tasks are accomplished by the two major components of the model, the Data Definition Language (DDL), and the Data Manipulation Language (DML).

The goal of the design process is to produce a specification that can be used to implement the database using a commercial DBMS. [Ref. 22]

The following three chapters will step through this entire process. The next step is to begin the user requirements analysis of the risk assessment and computer control environment at the Naval Postgraduate School. This will produce an enterprise model and and a description of the data elements. These products will then be used to perform the logical database design followed by the physical database design.

III. USER REQUIREMENTS ANALYSIS

The user requirements analysis phase, referring to Figure 2.1, involves four steps: problem recognition, evaluation, specification, and review. The first step, problem recognition, has already been covered in Chapter I. The next step, problem evaluation, involves analysis of the flow and structure of information to build user specifications.

A. USER ENVIRONMENT

OPNAVINST 5239.1A provides detailed guidance on risk assessment methodologies. Two methodologies are recommended. Methodology I is the more complex method and the standard for most ADP environments. It is this methodology that will be used for user requirements analysis and to produce an initial database system design for NPS.

Risk assessment methodology I can be broken down into five steps as shown in Figure 3.1. The first step is Asset Identification and Valuation. There are two basic parts to the initial step. First, a complete, up-to-date listing of all NPS ADPE assets is required. Second, the impact value for each asset must be determined for each applicable impact area.

A complete and accurate listing of computer assets is essential if the risk assessment is to be valid.

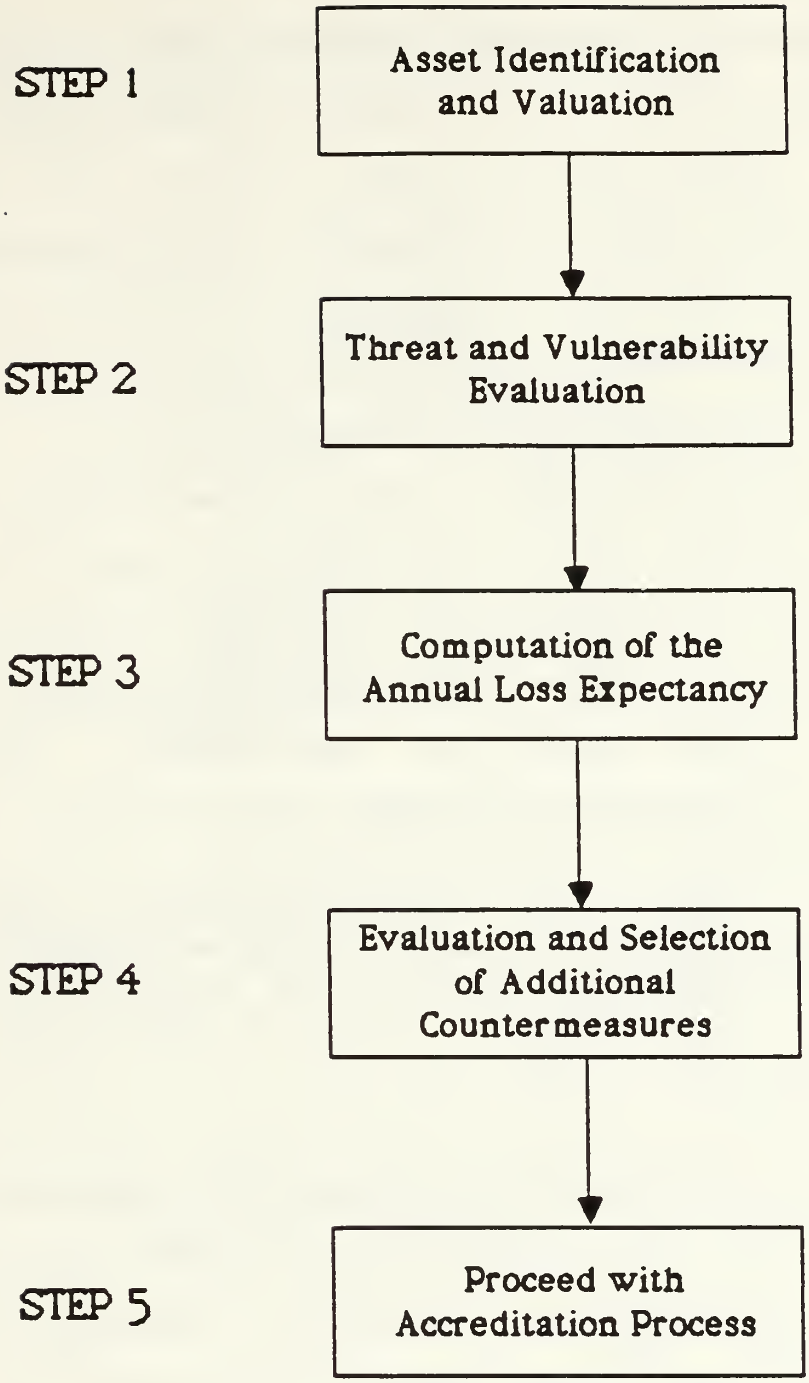


Figure 3.1 Major Steps of Method One Risk Assessment

Additionally. the inventory must be easily accessable by authorized users. Ease of access is important for several reasons. If the listing is easy to access. then it is more likely to be used and maintained. Any listing must be maintained if it is to be reliable. This is the goal of a risk assessment asset identification listing. This is also the goal of any asset control mechanism.

Part two of the asset identification process involves assigning impact value ratings once assets have been identified. The procedure is to determine dollar amounts for each of the four ways in which threats can impact an asset. The four impact areas are modification, destruction, disclosure and denial of service.

1. Modification refers to the value of software or data asset values based on the cost to correct the consequences of the modification and/or the cost of locating and recovering from the modification itself. The value of the assets should be based on the total cost to detect, locate, and correct the modification. [Ref. 23]

2. Destruction refers to the value. including the cost to reconstruct or replace the asset. as well as. the costs incurred from denial of service caused by the destruction of the asset.

3. Disclosure refers to the impact of disclosure of classified data.

4. Denial of service refers to the value of costs incurred from all denial of service, except for that caused by destruction. [Ref. 24]

To simplify the process. estimates are provided for impact and frequency. Dollar values and associated ratings

are scaled by a factor of ten. The sample range of impact values is shown in Table III.

TABLE III

Impact Value Ratings

Impact Value	Rating
$10	1
$100	2
$1.000	3
$10.000	4
$100.000	5
$1,000,000	6
$10.000.000	7
$100.000,000	8

Guidelines for Impact of Disclosure of Sensitive Data

For Official Use Only	$1.000
Privacy Act or Confidential	$10.000
Secret	$100,000
Top Secret	$1.000,000

This procedure requires the risk assessment team to list ADP assets and impact information on an Asset Valuation Worksheet. An example is shown in Figure 3.2.

NPS is presently in the process of conducting its first ADPE risk assessment. There are no established procedures or in situ data flows. Because of this, this thesis proposes that a relational database design be used to facilitate this required risk assessment procedure. Additionally, once

ASSET VALUATION WORKSHEET

1. ASSET NAME

System A Operating System and Support Programs

2. ASSET DESCRIPTION AND JUSTIFICATION OF SUPPORT PROGRAMS

Operating system and compiler support software for the System 'A' timesharing system.

Impact of modification was determined to be negligible. except in those cases where modification would result in denial of service. Those figures were included under denial of service.

Destruction was based on total destruction of all software and on-site backup tapes. These figures include denial of service caused by destruction.

Forty hours is required for delivery and check out of replacement O/S software. 75 users denied service at $12/hour: plus 6 system programmers at $14/hour for 16 hours; plus 3 data processing technicians at $8/hour for 36 hours. Total for the operating system - $36.936.

Sixty users denied use of the compiler at $12/hour for 24 hours; plus 1 system programmer at $14/hour for 8 hours. Total for the compiler support software - $1.832.

Reconstruction of compiler support data based on 618 hours to re-enter data at $8/hour. Total for compiler support data - $4.944.

Disclosure - N/A.

Denial of service was based on the number of users denied service for an average service outage.

Operating system: 35 users at $12/hour for 1 hour - $420.

Compiler support software: 35 users at $12/hour for .5 hours - $210.

Compiler support data - N/A.

3. SUCCESSFUL ATTACK FREQUENCY RATING BY IMPACT AREA.

☐ MODIFICATION	☐ DESTRUCTION	☐ DISCLOSURE	☐ DENIAL OF SERVICE
N/A	5	N/A	3

Figure 3.2 Asset Valuation Worksheet

developed, the relational database can also be used to control computer assets at NPS.

The number of computer assets at NPS has been growing rapidly over the past couple of years. Current accounting for computer resources is handled by several different departments. The different curricular departments maintain data, in the form of copies of equipment receipts, on most assets in their custody. The Computer Council is required to maintain an inventory of all hardware assets [Ref. 25]. A third point of control, the supply department, maintains purchase account data on most ADPE purchased through the Navy.

Clearly, there is not one reliable and easily accessible source available to provide answers to questions about ADPE resources. The following is a list of questions that department chairmen or the NPS Security Manager would like to be able to answer:

1. How much has NPS spent on computer equipment?
2. Who has custody of computer assets?
3. Where are computer assets located?
4. How much money has been spent on hardware? on software?
5. How many AST boards are there? Where are they?
6. How were the computer assets paid for?
7. How many modems does NPS have?
8. Is NPS providing enough security for its computer equipment?

9. What are the security risks to our computer equipement?

10. What are our top priorities relating to asset security vulnerability?

11. Where should NPS concentrate its effort to strengthen security?

This is just a sampling of questions that could be asked.

B. USER REQUIREMENTS

As previously discussed. the user must be able to keep an up-to-date inventory of all NPS computer assets. The primary objective of the proposed database system is to facilitate risk assessment. However, another application, closely related to the risk assessment objective is that of computer asset accountability for the purpose of control, in the sense of efficient and effective utilization. Therefore, user requirements for a dual purpose system will addressed.

The requirements for step one, of methodology I. in Figure 3.1 of the risk assessment procedure have already been defined in Chapter I. The remaining four steps should also be addressed during the user requirments analysis phase. Data in all five steps is closely linked together. In step two. Threat and Vulnerability Evaluation. each asset is analysed in terms of every threat it is subject to and the probability of occurance of each threat. Tables IV and V provide lists of threat frequency ratings and examples of generally recognized threats and their impacts.

The procedure involves identifying the threat that could cause the impact indicated on the Asset Valuation Worksheet for each asset.

TABLE IV

Successful Attack Frequency Rating

Frequency	Rating
Once in 300 years	1
Once in 30 years	2
Once in 3 years	3
Once every 4 months or 3 times a year	4
Once a week or 52 times a year	5
Once a day or 365 times a year	6
Once every 2 hours	7
Once every 15 minutes	8

Step three. of Figure 3.1. involves computation of the annual loss expectancy values (ALE). The impact dollar value ratings and the successful attack frequency ratings are used to produce the annual loss expectancy value for each of the four impact areas. This step provides a quantitative figure for each impact area and also the total ALE for the activity. Table VI shows annual loss expectancy computation. Samples of the ALE computation worksheet and risk assessment matrix are enclosed in Appendix A.

Step four. evaluation and selection of additional countermeasures, considers the evaluated countermeasures and

TABLE V
Threats and their Impacts

Threats	Impact Areas			
	Destruction	Disclosure	Modification	Denial of Service
Emanations/Eavesdropping	No	Yes	No	No
Emanations (Interference)	Yes	No	Yes	Yes
Alteration of software	Yes	Yes	Yes	Yes
Alteration/Failure of hardware	Yes	Yes	Yes	Yes
Unintentional operator error	Yes	Yes	Yes	Yes
Unintentional data entry error	Yes	Yes	Yes	Yes
Unintentional system programmer error	Yes	Yes	Yes	Yes
Unintentional disclosure	No	Yes	No	No
Misuse of computer resources	Yes	Yes	Yes	Yes
Power instability	Yes	No	Yes	Yes
Telecommunications failure	No	No	No	Yes
Environmental control failure	No	No	No	Yes
Natural disaster	Yes	No	No	Yes
Water damage (Internal/External)	Yes	No	No	Yes
Fire (Internal/External)	Yes	No	No	Yes
Enemy overrun/Civil disorder	Yes	Yes	No	Yes

selects which countermeasures to implement and in what priority. Priority is determined based on return on investment. Countermeasures that yield the highest return on investment are implemented first. As each additional countermeasure is implemented, it will affect the overall ADP security posture and ALE.

This procedure involves taking the threats with the highest potential for damage, identifing a countermeasure that could significantly reduce the vulnerability which these threats exploit, and then preparing an additional countermeasure evaluation worksheet.

Data items include return on investment, annual cost, original ALE saving, and countermeasures.

Step five involves the accreditation process. Pending implementation of all countermeasures, the designated approving authority (DAA) will grant the accreditation and issue interim authority to operate, or order operations to cease. See Appendix B for more information about the accreditation process.

A detailed discussion of these procedures can be found in OPNAVINST 5239.1A.

C. THE SYSTEMS FUNCTION HIERARCHY CHART

The third step of the user requirements analysis phase is specification. There are several methods available that can be used to specify the structure and flow of information in

the enterprise. Two methods that will be covered in this thesis are the systems function hierarchy chart and the BACHMAN diagram.

TABLE VI

Annual Loss Expectancy Computation

i = Impact Value Rating

f = Successful Attack Frequency Rating

For Impact. I = 10i

For Frequency F = 10f/3000

LOSS = IMPACT (I) X FREQUENCY OF OCCURENCE (f)

The systems functions hierarchy chart is a representation of the logical relationship among individual elements of data. This information structure is the blueprint for defining organization. methods of access. degree of associativity. and processing alternatives for information. This information structure can have a significant impact on data design requirements and therefore must represent the hierarchy of data in a readable. unambiguous manner. Figure 3.3 shows the system functions hierarchy chart for a computer risk assessment database. At the top level is a single block

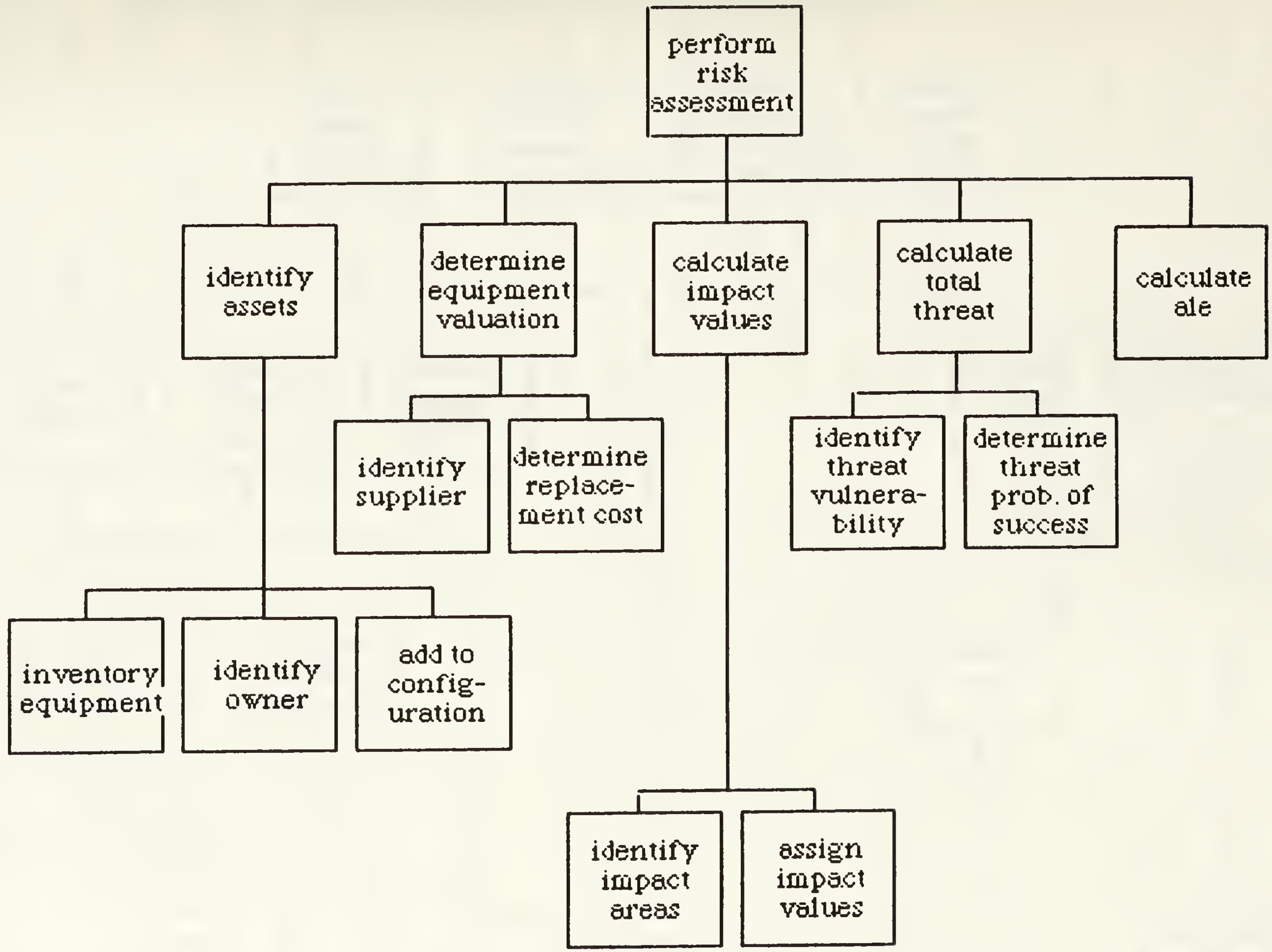


Figure 3.3 System Function Hierarchy Chart

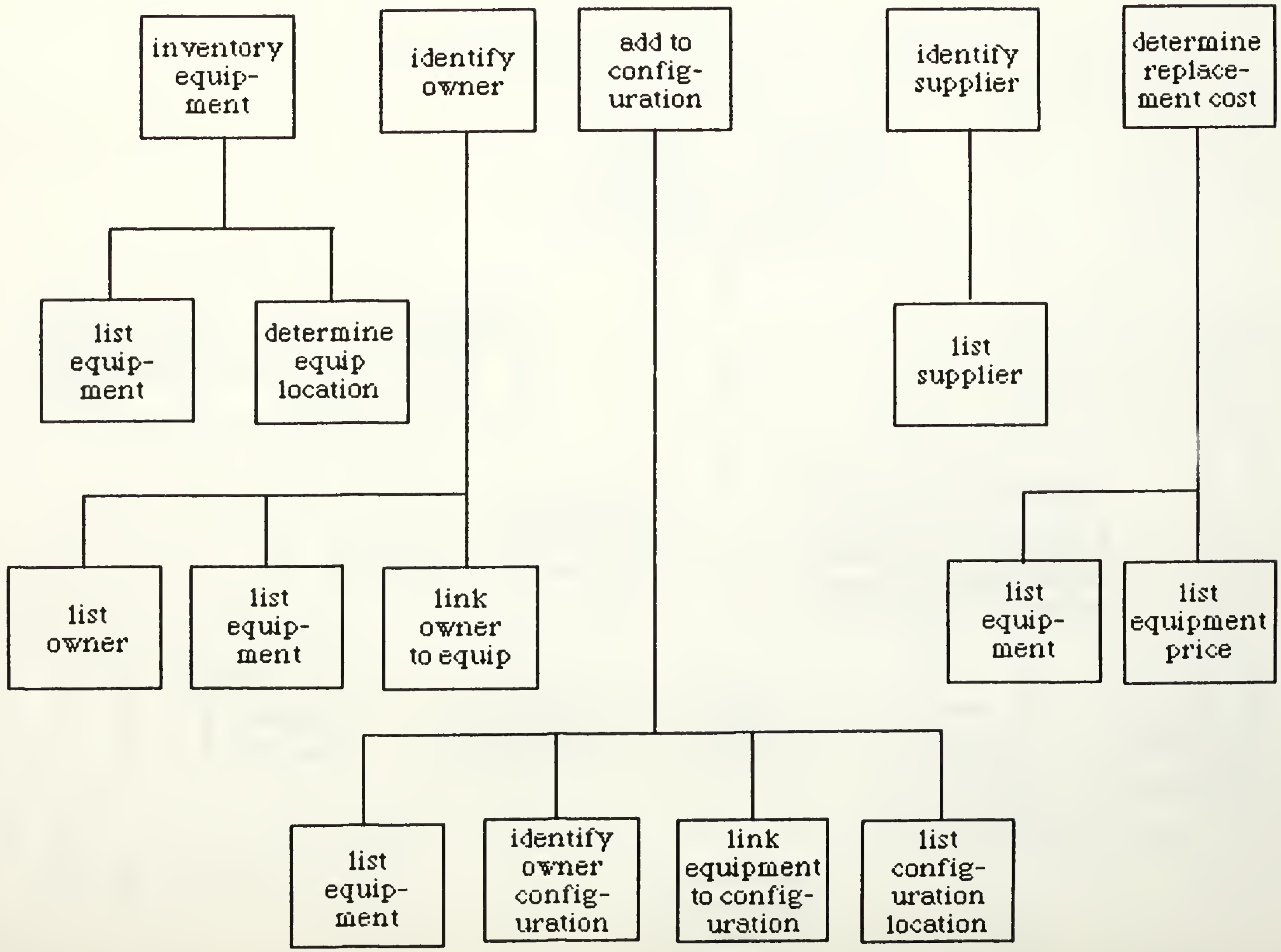


Figure 3.3 System Function Hierarchy Chart (cont'd)

that represents the entire hierarchy. Succeeding levels contain blocks that represent various categories of information that may be viewed as a subset of blocks further up the tree. At the lowest level, the diagram shows individual data entities. [Ref. 26]

D. THE ENTERPRISE MODEL

The enterprise model results in identification of the data items and the data relationships in the enterprise. This procedure provides a starting point in the design and data analysis process. To begin, it is necessary to define primitives in the real world. These real world primitives will be associated with conceptual primitives in the logical database design phase. These real world primitives are defined in Table VII.

1. Defining Data Elements

The foundations of data modeling begin with identification and understanding of fundamental structures or primitives in the real world.

The next step is to identify the elements of the enterprise in terms of developing a representation of the enterprise. This is an iterative process as is much of the database design process. The results of this iteration will merely provide a starting point to work from. Throughout the design process and even after the system is

implemented, problems will likely be identified which will necessitate redefining the data elements of the enterprise.

TABLE VII

Primitives in the Real World

Primitive	Definition
Object	Phenomena that can be represented by nouns.
Object Class	A group of objects formed by generalization.
Properties	Characteristics of objects.
Property Value Set	The collection of values that a given property may have.
Fact	The intersection of a given object with a given property value set.
Association	A connection of objects of the same or different classes.

Table VIII presents the enterprise object classes, objects and object descriptions. A list of examples of computer assets is provided by OPNAVINST 5239.1A.

2. <u>BACHMAN Diagram</u>

The BACHMAN diagram is one technique that can be used to specify the data relationships of an enterprise. The circles or bubbles represent objects or entities and the

TABLE VIII

Table of Object Descriptions

OBJECT CLASS	OBJECT	DESCRIPTION
HARDWARE	CPU	The central processing unit. including housing and chassis. Look at other object descriptions to see what other object descriptions are available. Note: If CPU is not detachable from other elements of the computer system. then unit will be logged under the CPU object type.
	MAIN MEMORY	Only external main memory units. Primarily associated with main frame computers.
	I/O CHANNEL	External units primarily associated with large computer systems.
	OPERATORS CONSOLE	Primarily associated with large computer systems.
	KEYBOARD	Detachable type keyboards.
	TERMINAL	Includes local and remote types.
	MONITOR	Detachable from other system elements.

TABLE VIII (cont'd)

MULTIFUNCTION BOARD	Specifically related to microcomputers. Defines the lowest level of asset identification for control of microcomputer hardware. These boards contain various numbers of memory chips. I/O ports, and speaker hardware.
DRUM	Secondary storage.
DISK DRIVE	Includes hard disk and floppy disk drives. internal and external.
DISK PACK	Portable; separate from drive.
DISKETTE	Floppy disks.
TAPE DRIVE	For micros to main frames. reel-to-reel and cassette.
MAGNETIC TAPE	Reel-to-reel only.
TAPE CASSETTE	Cassettes only.
CARD PUNCH	Card punch machine.
CARD READER	Card reading machine.
PUNCHED CARD	Punched cards.
PAPER TAPE	Punched paper tape.
PAPER TAPE READER	Paper tape reading machine.
NETWORK FRONTEND	Frontend processors.
DATABASE MACHINE	Specialized database processors.

TABLE VIII (cont'd)

	INTELLIGENT CONTROLLER	Identify controllers that are detached from main processors.
	PRINTER	Provide hardcopy printout.
SOFTWARE	OPERATING SYSTEM	Such as MS DOS. UNIX. CPM.
	APPLICATION PROGRAM	All specialized user programs.
	SYSTEM UTILITIES	Generally resident functional programs that are used to manipulate data and programs. Could be easily confused with application programs.
	TEST PROGRAM	Diagnostic program.
	COMMUNICATION	Specialized application program. Could be confused with application program.
COMMUNICATIONS	COMM LINES	
	COMM PROCESSOR	Specialized. detached hardware.
	MULTIPLEXOR	Detached.
	SWITCHING DEVICES	Detached.
	MODEM	All categories; micro to mainframe.

arrows represent the relationship between objects. Figure 3.4 shows the general form of data in the computer control and risk assessment project. The diagram is called a BACHMAN diagram, or a data structure diagram. The BACHMAN diagram only shows the relationships among records. The lines connecting the objects (entities) represent the relationship between the ojbects as one-to-one, one-to-many, or many-to-many depending on whether a line has single arrowheads on either end, double arrowheads on one end, or double arrowheads on both ends.

This BACHMAN diagram models the global risk assessment and computer resource control project.

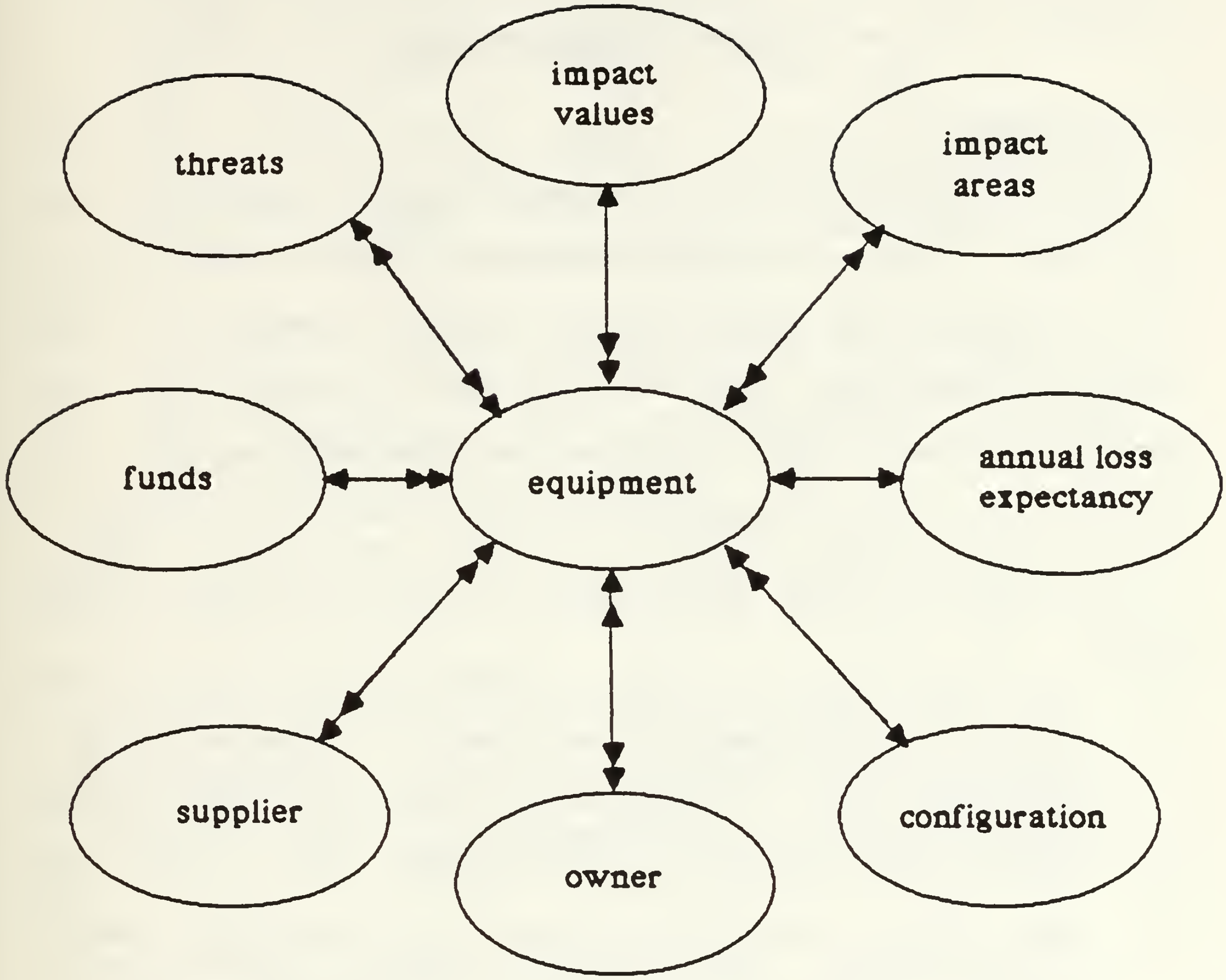


Figure 3.4 Bachman Diagram of the Enterprise Model for Computer Risk Assessment at NPS

IV. LOGICAL DATABASE DESIGN

A. PROCEDURE

The objective of the logical design phase in the relational database system development process (Figure 2.1). is to specify the logical format of the database. This involves identifying the records to be maintained, the contents of each record, and the relationships between the records. The resulting design is called the logical schema.

The logical design phase generally involves five design tasks.

1. The first task is identification of the data to be stored in the database. This step is accomplished by processing the data from the data dictionary and screening out information that will not be incorporated into the database, i.e.. descriptions of reports, screens, and input documents.

2. Next, it is necessary to standardize the terms used to name data. The failure to identify synonyms and aliases can severly degrade performance of a database. Worst of all, it can result in misinformation. Maintaining the integrity of the database is important and it is one big advantage of database systems over traditional file systems, if properly designed.

3. The third step is to define records and relationships. The process of defining records and relationships is largely intuitive [Ref. 27]. Records are defined by identifying the data items they will contain. Important considerations during this process include; designing flexible and expandable records that can evolve with the enterprise, designing effective record structures, and avoiding the using implied data in record definition.

Relationships during step three are defined in terms of how the users see them. The goal is to include all useful relationships in the logical schema. That is, specify all enterprise relationships that are needed by the user in practice.

4. The fourth step in logical database design is to define database processing. This requires analysis of how the database is manipulated to produce required results. This can be accomplished by using a method called transform analysis.

5. The final step is design review. The purpose of review is to identify problems with the design. At this point a decision on whether to continue or not has to be made. If the logical design is approved, the problems are corrected and the project proceeds to the physical design phase.

Now it is time to begin the logical design phase which will be developed based on information obtained from the user requirements analysis phase. The next section begins with a brief discussion of the Semantic Data Model which will be used during the logical design phase. The specific Semantic Data Model that will be used is the Entity-Relationship Model.

B. SEMANTIC MODELING

There are several database models available for the analysis and structure of data. However, these models are generally not suited to handle both logical and physical design problems. The relational model can be used for both phases of the design process, but it is not recommended.

There are several drawbacks to the use of the relational model for logical database design. It is too detailed to be used in the initial stages of design, and it lacks the

semantic structure to make unambiguous choices in modeling enterprises [Ref. 28]. The relational model emphasizes functional dependencies for defining the semantic nature of data. This process is usually too complicated during the initial stages of database design.

Semantic models are one way to integrate the relational model into the systems development.

The goal is to provide abstractions that naturally adapt to the way that users describe enterprises [Ref. 29]. The semantic model is used to identify the essential enterprise constructs and then is modified by applying relational criteria. The semantic model uses similar abstractions for defining. naming. and classifying object sets and associations as listed in Table IX.

There are as many different semantic models as there are varieties of ways to represent modeling abstractions. One well known example is the EntityRelationship Model.

C. THE ENTITY-RELATIONSHIP MODEL

The entity-relationship model is based on three main semantic concepts: entities. relationships. and attributes. The enterprise must be described in these three terms. See Table X for a description of these three terms. Entities describe things in the enterprise which in turn are described by attributes. Entities can also interact with one another in any number of relationships. [Ref. 30]

TABLE IX

Tabular Description of Enterprise

OBJECT CLASS	PROPERTIES
HARDWARE	TYPE NAME MANUFACTURER COST DATE ACQUIRED OWNER
SOFTWARE	TYPE NAME MANUFACTURER COST DATE ACQUIRED OWNER
COMMUNICATIONS	TYPE NAME MANUFACTURER COST DATE ACQUIRED OWNER
OWNER	NAME ADDRESS PHONE NUMBER
CONFIGURATION	NUMBER OWNER LOCATION
SUPPLIER	NAME ADDRESS PHONE NUMBER
FUND	TYPE NUMBER SPONSER

TABLE X

Terminology

1. Entities - distict objects within a user enterprise
2. Relationships - meaningful interactions between objects
3. Attributes - describe the entities and relationships

Entities and relationships are organized into sets or classes. Entities or relationships in a set have the same attributes. These sets all have unique names.

1. Entity-Relationship Diagram

Entities and Relationships are represented diagramatically with the entity-relationship diagram. Entity sets are represented by rectangular boxes and relationships are represented by diamond-shaped boxes. The relationship boxes are joined to the entity boxes that participate in the relationship.

Figure 4.1 shows the entity-relationship diagram for the complete conceptual risk assessment database system. The entity-relationship diagram for the control and asset identification view of the database is shown in Figure 4.2. This view was constructed using the data element information and data relationships developed during the data analysis stage.

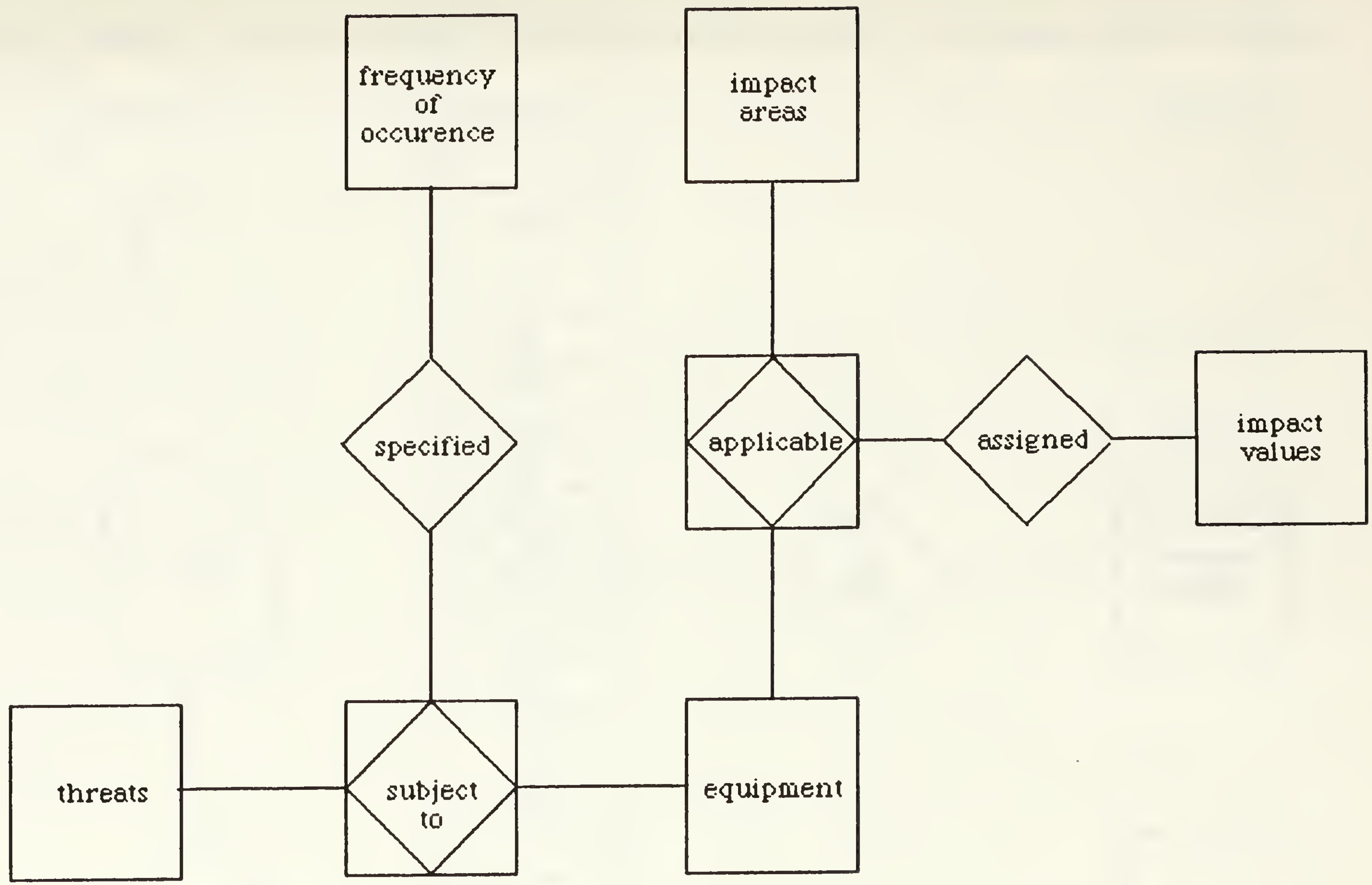


Figure 4.1 Conceptual Entity-Relationship Diagram of Risk Assessment Database Design

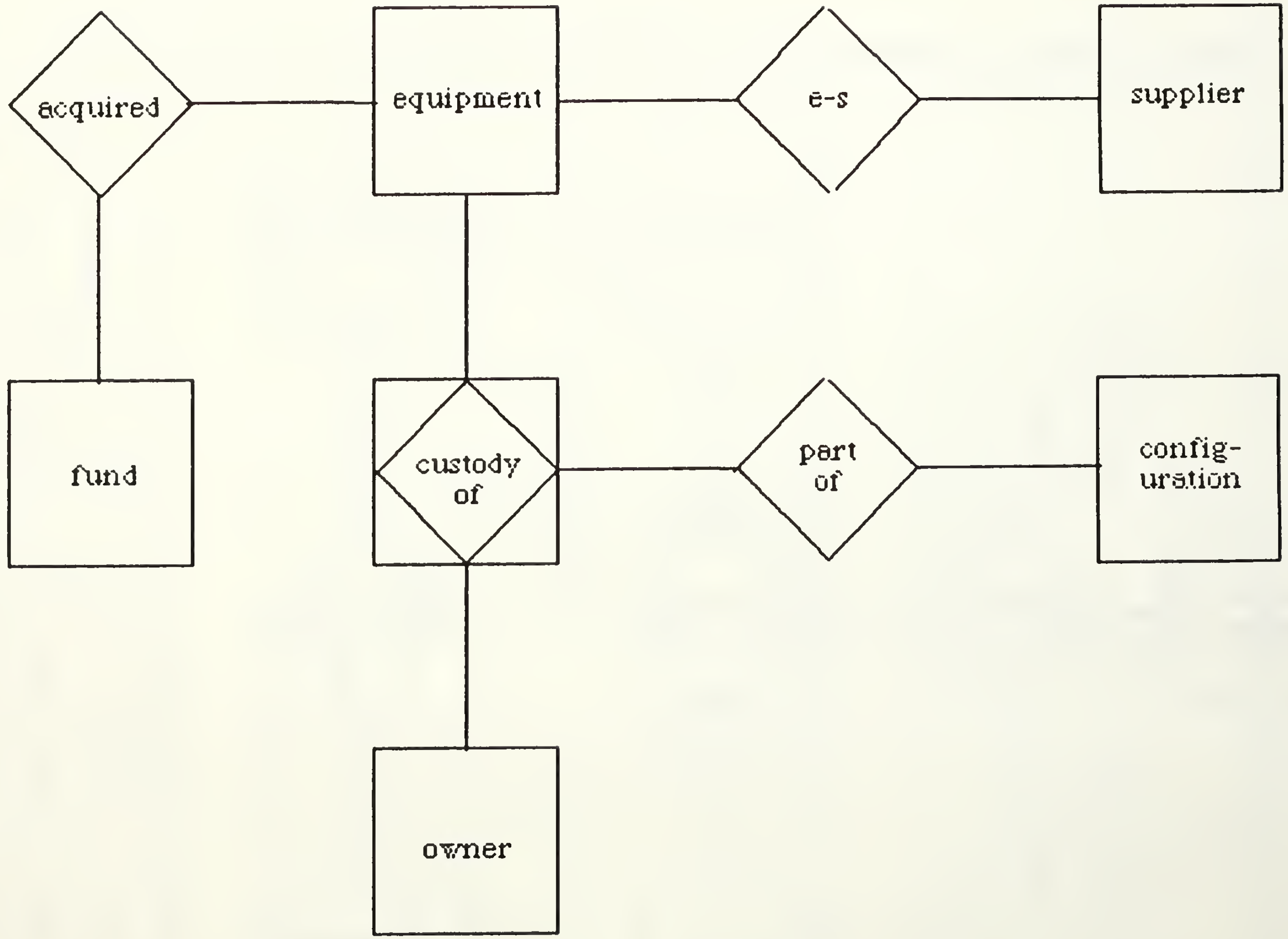
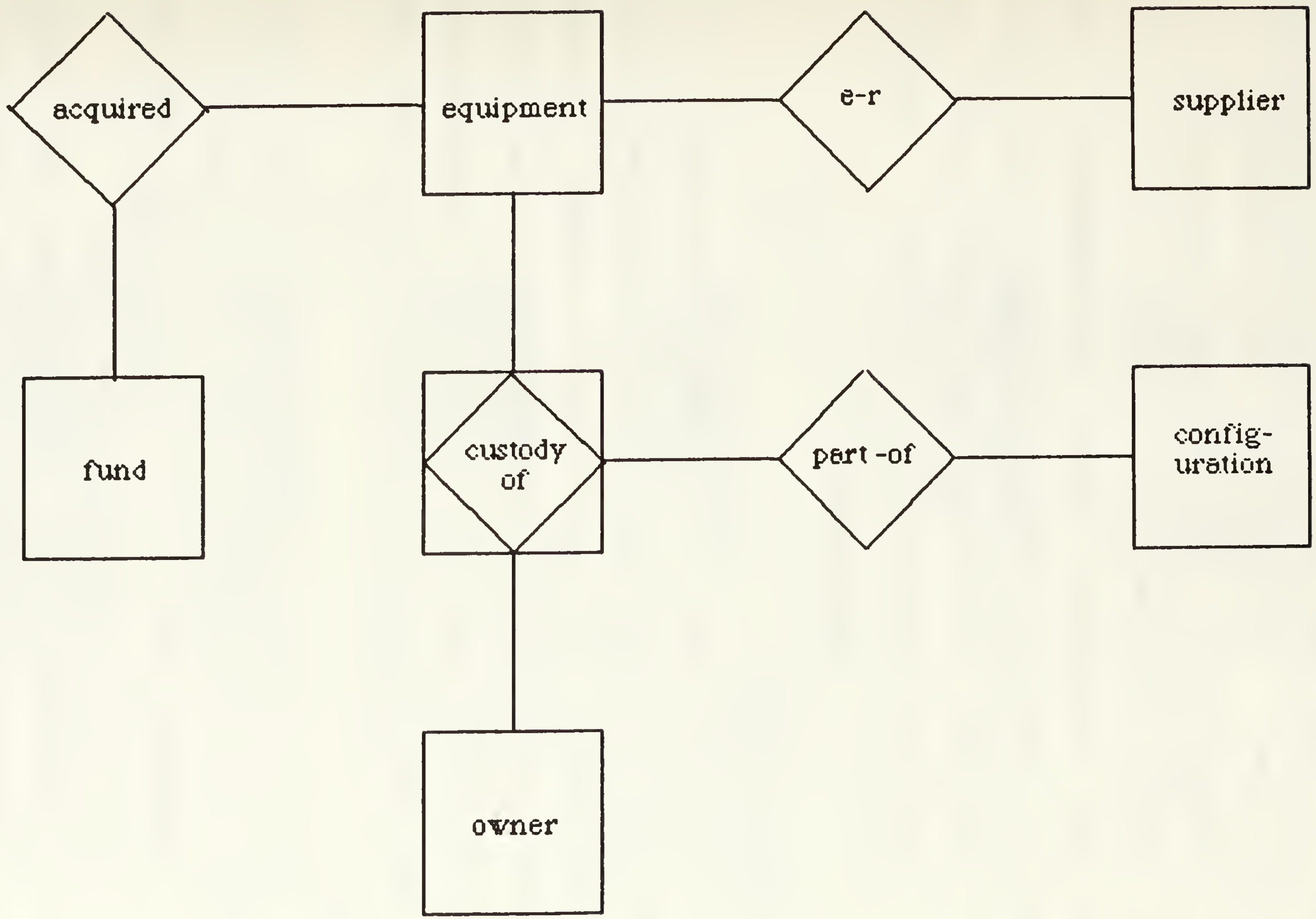


Figure 4.1 (cont'd) Conceptual Entity-Relationship Diagram of Risk Assessment Database Design



Enitiy-Relationship Diagram of Asset Identification View of Risk Assessment Database System

Figure 4.2

Table XI shows the attributes for each entity and relationship type. Attributes describe some aspect of an entity or relationship. Some attributes are also used to uniquely identify each entity occurence. One or more attributes can be used to identify an occurence. These attributes are called key attributes. Relationships must be uniquely identified in the same way. Relationships are normally identified by more than one attribute. Tentative keys are indicated by underlining.

2. Logical Schema

The logical schema is composed of the records to be maintained, their contents, and the relationships among those records specified. In the entity-relationship diagram, entities and relationships become records. The entity-relationship diagram shows the relationships between records. however, the records and their contents have not yet been identified.

The process of logical record structure involves making each entity and relationship in the entity-relationship diagram a separate record. The attributes associated with each entity and relationship then become fields.

As the requirements are evaluated and the design progresses, constraints on data items will be identified. Constraints are limitations on the values that database data can have. There are three common types of constraints:

TABLE XI

Entity-Relationship Attribute List

EQUIPMENT

EQUIPMENT-NUMBER
EQUIPMENT-TYPE
MANUFACTURER
EQUIPMENT-NAME
COST
DATE-ACQUIRED

CUSTODY-OF

EQUIPMENT-NUMBER
OWNER-NUMBER
CUSTODY-NUMBER

OWNER

OWNER-NUMBER
OWNER-NAME
OWNER-ADDRESS
OWNER-PHONE

PART-OF

CUSTODY-NUMBER
CONFIGURATION-NUMBER

CONFIGURATION

CONFIGURATION-NUMBER
OWNER-NUMBER
CONFIGURATION-LOCATION
CONFIGURATION-ADDRESS
CONFIGURATION-PHONE

TABLE XI (cont'd)

Enitity-Relationship Attribute List

E-S

SUPPLIER-NUMBER
EQUIPMENT-NUMBER
EQUIPMENT-PRICE

SUPPLIER

SUPPLIER-NUMBER
SUPPLIER-NAME
SUPPLIER-ADDRESS
SUPPLIER-PHONE

E-F

EQUIPMENT-NUMBER
FUND-NUMBER

FUND

FUND-NUMBER
FUND-TYPE
SPONSER

field constraints. inter-record constraints and intra-record constraints. Field constraints limit the values that a given data item can have. Interrecord constraints limit values between fields in different records. Intrarecord constraints limit values between fields within a given record. [Ref. 31]

See Table XII for record specifications of the logical schema.

D. PROBLEMS WITH SEMANTIC MODELING

When constructing the entity-relationship model. the design should arguably consider the correspondence between the semantic model to normal relations. Hovever. imposing such constraints on the semantic process can restrict the naturalness of the model. which is the reason it is used.

Semantic models do not possess the criteria necessary to prevent anomalies and eliminate redundancies. This role is reserved for the relational model in the next stage.

Another problem is the treatment of relationships in the entity-relationship model. The entity-relationship model can easily accomodate one-to-one or many-to-many relationships. Additionally. the entity-relationship model can be defined on a single entity-type as well as on three or more types. However. most DBMS's are not as flexible.

TABLE XII

Record Specification for Logical Schema

Field	Description
EQUIPMENT Record	
Equipment-number	Numeric. 6 decimal digits
Equipment-type	Alphanumeric, 40 characters
Manufacturer	Alphabetic. 30 characters
Equipment-name	Alphanumeric. 30 characters
Cost	Alphanumeric. 10 characters
Date-acquired	Format: YYMMDD
CUSTODY-OF Record	
Equipment-number	Numeric. 6 decimal digits
Owner-number	Format: 999:99:9999
Custody-number	Numeric. 8 decimal digits
OWNER Record	
Owner-number	Format: 999:99:9999
Owner-name	Alphabetic. 30 characters
Owner-address	Alphanumeric. 40 characters
Owner-phone	Format: (999)999-9999
PART-OF Record	
Custody-number	Numeric. 10 decimal digits
Configuration-number	Numeric. 5 decimal digits

TABLE XII (cont'd)

Fecord Specification for Logical Schema

Field	Description
CONFIGURATION Record	
Configuration-number	Numeric. 5 decimal digits
Owner-number	Format: 999-99-9999
Configuration-location	Alphanumeric. 30 characters
Configuration-address	Alphanumeric. 30 characters
Location-phone	Format: (999)999-9999
E-S Record	
Supplier-number	Numeric. 8 decimal digits
Equipment-number	Numeric. 6 decimal digits
Price	Alphanumeric. 10 characters
SUPPLIER Record	
Supplier-number	Numeric. 8 decimal digits
Supplier-name	Alphanumeric. 30 characters
Supplier-address	Alphanumeric. 30 characters
Supplier-phone	Format: (999)999-9999
E-F Record	
Equipment-number	Numeric. 6 decimal digits
Fund-number	Numeric. 4 decimal digits
FUND Record	
Fund-number	Numeric. 4 decimal digits
Fund-type	Alphabetic. 30 characters
Sponser	Alphanumeric. 30 characters

Additionally. although the process of defining entities and relationships appears to be quite clear. the database designer must decide how to assign entities and relationships. It is up to the designer to establish the importance of various objects in the organization being modeled. Consequently. the design process is not deterministic. Different designers can produce different models of the same enterprise and which in turn produce totally different results. [Ref. 32]

V. PHYSICAL DATABASE DESIGN

A. OBJECTIVE

Before starting the physical design process it is necessary to take a look at the system that will be used to implement the database. One requirement of the design specification is that it should be easily converted to the implementation model. The data design model is dependent on the DBMS. In this case, it was determined at the outset. that the database would be processed by a relational DBMS. Therefore, the relational database model will be used to express the design of the database.

There are two basic steps in the physical database design process. The first step involves transforming the logical schema into the particular data constructs to be used by the particular DBMS selected. The first step will produce detailed specifications of the database that will be used during database implementation to write source statements that define the database structure to the DBMS [Ref. 33]. The second step is to review the design and to modify it to achieve optimal performance.

B. THE RELATIONAL MODEL

The relational database model uses a single construct to represent both entities and relationships. The construct is

a two-dimensional table called a relation which is formally defined as an unordered set of ordered n-tuples. An n-tuple is a set of n values which is like a record. The values within a tuple have to be ordered because the values do not carry identifying labels. The columns correspond to attributes and each row is an occurence. A summary of relational model terminology can be found in Table XIII.

TABLE XIII

Relational Model Terminology

1. Relation - a two dimensional table that has several properties
2. Attributes - the columns of a relation
3. Tuples - the rows of a relation
4. Domain - the set of values that an attribute can have

A relation cannot contain any duplicate rows. It is important to note, however, that no relational DBMS enforces this constraint. The reason for preventing preventing duplicate rows is so that one collection of attributes will uniquely identify a tuple. The number of attributes that are needed, can range from one to all the attributes in the tuple. Each collection of attributes is called a candidate key, and one of these must

be selected as the primary key. Keys are used during the design process to avoid designs that have undesireable properties.

One reason data definition if so important in the relational model is that relationships are not represented by explicit links as they are in other models, but are carried in the data. Basic relational operations operate on entire collections of entities and relationships instead of dealing with them individually. In using a relational DBMS, the user specifies what is wanted, and the system must decide how to do it. For this reason, relational systems appear simple and easy to use. It is also conceptually easier to implement a relational system. However, a simple straight forward approach, without an understanding of the mathematical theory on which the relational model is based, can result in poor performance.

C. NORMALIZATION

The process of normalization invloves converting an arbitrary relational database design to one that avoids certain anomalies. The purpose is to produce a database design that can be manipulated in a powerful way with a simple collection of operations while minimizing data anomalies and inconsistencies.

With some relations, changing data can have unexpected and undesireable consequences. These are known as

modification anomalies. Two common modification anomalies are insertion and deletion anomalies. Insertion anomalies result when information is gained about two different entities with a single insertion. Deletion anomalies result when facts are lost about two entities with one deletion. Consider the ACTIVITY relation in Figure 5.1. It has the attributes STUDENT_ID, ACTIVITY, and FEE. The relation contains information about what activity a student engages in and how much it costs.

RELATION: ACTIVITY

STUDENT_ID	ACTIVITY	FEE
200-22-1234	TENNIS	50
111-23-2345	SKIING	100
752-41-4982	GOLF	75
543-65-3856	SWIMMING	60

Figure 5.1 Relation ACTIVITY

The fee is dependent on the activity and is the same for all students engaging in that activity. In we deleted the tuple for the first student. 200-22-1234 in Figure 5.1. we loose the fact that student 200-22-1234 is a tennis player. However. we would also loose the fact that tennis costs $50. This is an example of a deletion anomaly. If we wanted to add an activity to the relation. for example. racketball. which costs $55, we could not do so until a student enrolls in that activity. This is an example of an insertion anomaly. These anomalies can be prevented by a process called decomposition where a single relation is broken down into two separate relations. To ensure reliability. data integrity. and efficiency. modification anomalies must be eliminated. Anomalies can be eliminated by changing the database design. Relations can be independent or interdependent. Usually, the less interdependency, the better.

The problem of modification anomalies focused attention on the form of the relations that resulted in data errors. The problem was addressed by the concept of relational normal forms. There are seven different normal forms. in hierarchical order ranging from the First Normal Form (1NF). which includes all relations, to Domain Key/Normal Form (DK/NF). in which relations are free from all modification anomalies regardless of their type. Table XIV shows the seven different normal forms.

The first, second, third and Boyce-Codd normal forms all address anomalies caused by inappropriate functional dependencies. A functional dependency is a relationship between attributes. "Attribute Y is said to be functionally dependent on attribute X if the value of X determines the value of Y." [Ref. 33] Functional dependencies are denoted by the form, X --> Y, where attribute X is called the determinant, and Y is called the dependent variable.

Determinants may or may not be unique. Functionally dependent attributes need not be unique either. Functional dependencies can involve groups of attributes, and one or more attributes can determine several attributes.

TABLE XIV [Ref. 35]

Summary of Normal Forms

Form	Defining Characteristic
1NF	Any relation
2NF	All no-key attributes are dependent on all of the keys
3NF	There are no transitive dependencies
BCNF	Every determinant is a candidate key
4NF	Every multivalued dependency is a functional dependency
5NF	Join dependencies are satisfied
DK/NF	All constraints on relations are logical consequences of domains and keys

A key of a relation comprises one or more attributes that functionally determine or identify a tuple. Because a key functionally determines the entire tuple, it must be unique. If it were not unique, then the entire tuple would be duplicated, and this is not allowed by definition. [Ref. 36]

For a relation to be in second normal form, all nonkey attributes must be dependent on all the key attributes. A relation is in third normal form if it is in second normal form and has no transitive dependencies. Finally, a relation is in Boyce-Codd normal form if it is in third normal form and every determinant is a candidate key.

Even with a relation in Boyce-Codd normal form, anomalies can still arise from multivalued dependency. Additionally, anomalies can result from joining two projections on a relation. These two anomalies are eliminated by the fourth and fifth normal forms respectively. The DK/NF eliminates all modification anomalies. A relation is in DK/NF if every constraint on the relation is a logical consequence of the definition of keys and domains.

There are three criteria that should be used to produce an effective relational database design. They are the elimination of modification anomalies, relation independence and ease of use.

The first criteria, elimination of modification anomalies, will be achieved if the relation is put into

DK/NF. The second criteria is to design relation independence to support error free operation of the database system. The third criteria is to create a relational design that is easy to use. This involves trying to structure the relations so that they are familiar and seem natural to users.

Often, these three design criteria conflict with each other. It is the responsibility of the designer to assess priorities and make the best possible compromise because of the requirements. There are no rules for the questions of priority. [Ref. 37]

D. THE RELATIONAL SCHEMA

To translate a model into an operational system, the model has to be described in a form which lends itself to implementation. This initial model is called the relational schema and the language used to describe it is called the schema language.

One objective of the database system is to systematize the access to data elements. To be able to fetch data elements, irrespective of the file and record structure, we will have to provide descriptions which allow determination of position and type of the data elements by attribute name and value, or by attribute relationship. [Ref. 38]

This is an important element of obtaining a description of the database requirements. Defining the data elements

further. is necessary to generate the processing programs that deal with the database.

Data definitions and relationships are taken from the results of the previous design phases. Any new procedures that are identified at this stage are also included in the design.

An essential part of defining data elements is that the data elements must be defined in terms of programming language specifications. The process involves assigning a name and defining specific characteristics of data elements. This specification is known as the data type.

There are both quantitative and qualitative characteristics of data elements. Quantitative characteristics include name, type, domain and length.

Names have been used during the entire database development process to define attributes in records and relations. The name given to a data element is usually controlled by straight-forward rules and depends on the programming language used. These rules generally allow for a short, variable-length string of alphabetic and numeric characters, the first character being restricted to be alphabetic. Names used in files and databases are generally global. In a database, the name of a data element is affected only by structural scope, defined as the database or relation that contains the data element. [Ref. 39]

For example. the name for the data element, equipment identifier, is E:NUM.

A type specification is usually associated with each data element name. The type specification limits the values to be associated with the data-element names to a specified domain, simplifies processing by implying the category of legal operations to be used in the transformation of data and it provides specifications for encoding of the data values. [Ref. 40] Most computer languages provide a small number of general choices, such as, CHARACTER, DECIMAL, BINARY INTEGER, BINARY FIXED POINT NUMBER, and REFERENCE just to name a few. The data element E:NUM is specified as being NUMERIC.

Domain defines the range of allowed values for a data element. It can be valuable by keeping improper elements out of the database. For each data element then, there is a domain for which there is a corresponding domain definition. The attribute domain for the data element E:NUM is EQUIP_IDENTIFIERS which is defined as NUMERIC 999999. Each nine in this example represents a space for a numeric character. This number represents the plant account number when it is available.

Length can be specified as either fixed or variable. Another way to specify the length of the data element E:NUM is: E:NUM NUMERIC (6). where the number 6 specifies the length of the data element.

The complete data element definition for the data element E:NUM is illustrated in Figure 5.2.

NAME	E:NUM
TYPE	NUMERIC
DOMAIN	EQUIPMENT_IDENTIFIERS
LENGTH	(6)

Figure 5.2 Data Definition of E:NUM

Qualitative characteristics of data elements are used to provide additional semantic information. These include title, unit specification, essential data, undefined values, transformations, access privilege, and file management.

A title is sometimes used to provide a more detailed description of the data element. Unit specification is used to ensure standardization when units of measure are involved. Essential data refers to whether the data element is required or is optional in a record. Undefined values provide information about whether data can be missing or left undefined. Programs that operate on the database must be able to recognize this fact.

Transformations are important if there is a need to transform data between the outside world and the database. Access privilege defines which users are allowed access to which data elements. Finally, file management information may be contained in the schema. File management concerns matters such as control indexing, transposition, control of integrity, archiving, and erasure cycles.

The relation definitions of the risk analysis project are shown in Table XV. Attribute domains and domain definitions are listed in Tables XVI and XVII.

E. DATA MANIPULATION

In relational databases, relations are viewed as collections of records (tuples), and relational operations apply to entire relations. At the most elementary level, there are three operations that need to be discussed: projection, selection, and join. [Ref. 41]

The first relational operation, projection, produces a new relation that contains only the columns (attributes) from the original relation that are specified in the projection request. It is important to note that this operation will eliminate any duplicate tuples that may naturally occur.

The selection operation will produce only the rows (tuples) from the original relation that satisfy a certain condition or conditions. The results of both projection and

TABLE XV

Relation Definitions

1. EQUIPMENT (E:NUM, E:TYPE, E:MFG, E:NAME, E:COST, E:DTEACQ)

2. E-O (E:NUM, O:NUM, C:NUM)

3. OWNER (O:NUM, O:LNAME, O:FNAME, O:STRET, O:CITY, O:STATE, O:ZIP, O:PHONE)

4. O-C (C:NUM, CO:NUM)

5. CONFIGURATION (CO:NUM, O:NUM, CO:LOC, CO:STRET, CO:CITY, CO:STATE, CO:ZIP, CO:PHONE)

6. E-S (S:NUM, E:NUM, PRICE)

7. SUPPLIER (S:NUM, S:NAME, S:STRET, S:CITY, S:STATE, S:ZIP, S:PHONE)

8. E-F (E:NUM, F:NUM)

9. FUND (F:NUM, F:TYPE, F:SPONSR)

TABLE XVI

Attribute Domains

Attribute	Domain
1. E:NUM	EQUIP_IDENTIFIERS
2. E:TYPE	EQUIP_TYPES
3. E:MFG	MFG_NAMES
4. E:NAME	EQUIP_NAMES
5. E:COST	PRICES
6. E:DATACQ	DATES
7. O:NUM	OWNER_IDENTIFIERS
8. C:NUM	CUSTODY_IDENTIFIERS
9. O:LNAME	L_NAMES
10. O:FNAME	F_NAMES
11. O:STRET	STREET_ADDRESSES
12. O:CITY	CITY_NAMES
13. O:STATE	STATE_NAMES
14. O:ZIP	ZIP_CODES
15. O:PHONE	PHONE_NUMBERS
16. CO:NUM	CONFIG_IDENTIFIERS
17. CO:LOC	LOCATION_NAMES
18. CO:STRET	STREET_ADDRESSES
19. CO:CITY	CITY_NAMES
20. CO:STATE	STATE_NAMES
21. CO:ZIP	ZIP_CODES
22. CO:PHONE	PHONE_NUMBERS
23. S:NUM	SUPPLIER_IDENTIFIERS
24. PRICE	PRICES
25. S:NAME	SUPPLIER_NAMES
26. S:STRET	STREET_ADDRESSES
27. S:CITY	CITY_NAMES
28. S:STATE	STATE_NAMES
29. S:ZIP	ZIP_CODES
30. S:PHONE	PHONE_NUMBERS
31. F:NUM	FUND_IDENTIFIERS
32. F:TYPE	FUND_TYPES
33. F:SPONSR	SPONSER_NAMES

TABLE XVII

Domain Definitions

Domain Name	Format and Meaning
1. CITY_NAMES	CHAR (25)
2. CONFIG_IDENTIFIERS	numeric 99999; uniquely identifies each configuration
3. CUSTODY_IDENTIFIERS	numeric 9999999999; identify specific ownership of each piece equipment
4. DATES	numeric YYMMDD
5. EQUIP_IDENTIFIERS	numeric 999999; uniquely identifies each piece of equipment
6. EQUIP_NAMES	CHAR (30); standard commercial nomenclature
7. EQUIP_TYPES	CHAR (40); must fit one of the object categories listed is Table VII
8. F_NAMES	CHAR (10); first names of people
9. FUND_IDENTIFIERS	numeric 9999; uniquely identifies each fund
10. FUND_TYPES	CHAR (30); identifies fund as being research, OPN, O&MN
11. LOCATION_NAMES	CHAR (30); identifies location of configuration beyond street address - building name and room, home
12. L_NAMES	CHAR (20); last names of people
13. MFG_NAMES	CHAR (30); names of manufacturers

TABLE XVII (cont'd)

Domain Definitions

Domain Name	Format and Meaning
14. OWNER_IDENTIFIERS	999:99:9999; social security numbers are used to uniquely identify individual owners
15. PHONE_NUMBERS	(999)999-9999
16. PRICES	numeric 999999999.99
17. STATE_NAMES	CHAR (2); two character state abreviations
18. STREET_ADDRESSES	CHAR (30)
19. SUPPLIER_IDENTIFIER	numeric 99999999; uniquely identifies equipment suppliers
20. SUPPLIER_NAMES	CHAR (30); names of equipment suppliers
21. ZIP_CODES	numeric 99999; five digit postal zip code

selection are also relations and therefore can be used to produce any desired subset of an original relation.

The join relational operation is used to combine data from two relations into one. It produces a new relation that contains all the columns from both of its input relations. Tuples from the two input relations are combined by looking for matching values in a specified common attribute of each.

Most DBMS's, however, enable users to handle data manipulation at a higher level than the examples just described. DBMS's generally use English-like syntax and tend to hide the actual structure of the database from the user.

To process relations with a computer, it is necessary to have a clear, unambiguous language for expressing what you want to do. There are four different strategies for relational data manipulation: relational algebra, relational calculus, transform-oriented, and graphic. Since many DBMS products use transform-oriented relational language, this language will be emphasized.

Transform-oriented languages are a class of non-procedural languages that use relations to transform input data into desired outputs. These languages provide easy-to-use structures for expressing what is desired in terms of what is known. [Ref. 42]

An example of a relational Database Manipulation Language (DML) which is transform-oriented is SQL. SQL stands for Structured Query Language. The basic construct of

SQL is a mapping, whose syntax takes the for

```
SELECT <attribute>
FROM   <relation>
WHERE  <condition clause>
```

The simplest condition clause appears as:

```
<attribute>  <binary operator>  <value>
```

The binary operators include =, NEQ, $\geq$, $\leq$, >, <.

The output from a mapping is a set of values. The values are chosen by selecting each relation row that satisfies the condition clause. The value of the attribute of each such selected row becomes part of the output. [Ref. 43]

This chapter has presented some of the important aspects of the physical database design phase. It stressed the fact that all relational database designs are not equal. Some relational schemas suffer modification anomalies, some have unacceptable dependencies among relations, and some are poorly suited to users. The criteria for good relational designs are reduction or elimination of modification anomalies, minimization of interrelation constraints, and ease of use. An initial relational database design has been created for asset identification and valuation for risk assessment at NPS.

VI. IMPLEMENTATION CONSIDERATIONS

A. DATABASE MANAGEMENT SYSTEMS

The database management system (DBMS) is a specialized piece of software that is used to implement a database. The DBMS serves as an interface between the data and the user. How the database management system interfaces with the overall system is illustrated in Figure 6.1. Users and their application programs make requests to the DBMS. If the request is valid. the DBMS takes responsibility for performing the necessary database manipulations. However, it is not able to do this directly and must invoke the file handling capabilites of the operating system to read or write data. This interaction with the system introduces additional overhead. [Ref. 44] In currently available systems DBMSs appear to the operating system as just another user program.

There are two primary functions of a DBMS. First. it assists users in manipulating the database. and secondly. it protects the database from the users. Assistance is provided to users by program modules that perform standard functions such as data retieval or modification. Additional assistance is provided by program modules that perform functions on the database without the need for any programs to be written at all. Examples include query processors and report

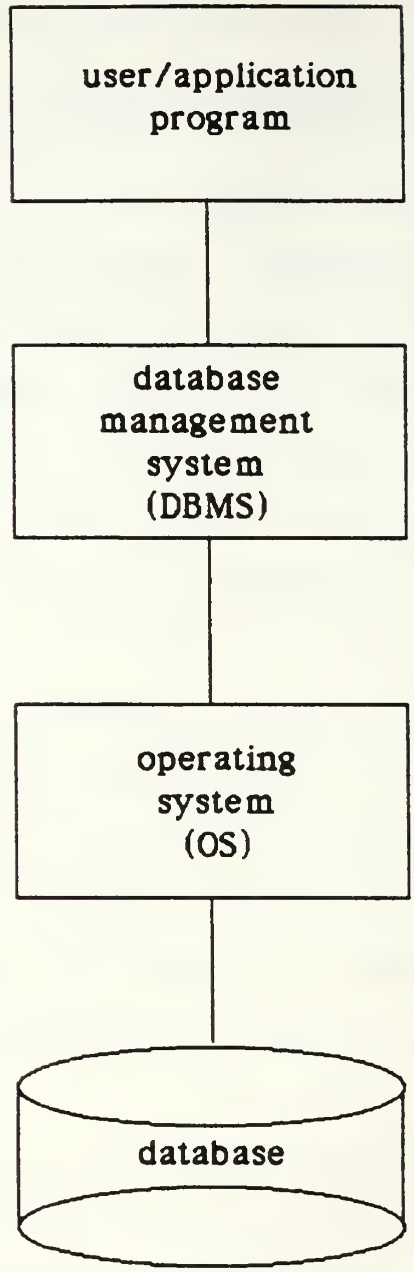


Figure 6.1 Relationship between DBMS and OS

generators. The use of standard system modules greatly enhances database system operations. It reduces the amount of work that must be done to implement new applications and also increases the reliability of applications. The DBMS provides a natural interface for user data. The interface must be independent of any physical storage structures. Finally, the DBMS should have the ability to construct different views of the database. That is, portions of the database that are irrelevant to an application can be hidden from it, and the structure of what remains can be adjusted to fit the application's specific requirements [Ref. 45].

The second function of a DBMS, database protection, is implemented primarily through a gate-keeping function for the DBMS. All user requests must be made through the DBMS. This allows the DBMS to evaluate each request and decide whether it should proceed. The decision can be made on both authorization criteria and on integrity criteria. A type of access control is also implemented through defining views to include or exclude particular portions of the database. [Ref. 46]

The above fundamental capabilities of a DBMS required to support the database are summarized in Table XVIII.

Two central functions of DBMS software are to define a database and to access the defined database. DBMSs use special languages to define databases. After a database is designed, the database structures are defined using the Data

Definition Language (DDL). Software to access databases is classified into three categories that coincide with three levels of users. The professional programmer, who developes programs for other users, employs embedded database access commands in a programming language. These commands are known as data manipulation language (DML). Most systems also provide query languages for the second level of user, the nonprogrammer user. A few systems provide a natural language interface for the third level of user, the casual user. [Ref. 47]

TABLE XVIII

Fundamental Capabilities of DBMS

1. Must provide a natural interface of user data
2. The interface must be independent of any physical storage structures
3. Different users should be able to access the same database, using different views of the database
4. Changes to the database can be made without affecting programs that make no use of the change

There is a subtle difference between data models and their implementation. Data models are abstractions while implementations are a software realization of the data model abstraction. There are a variety of DBMSs, and each is

implemented by different software structures. These software structures are called DBMS architectures. The software components of a DBMS must provide for each of the functions listed in Table XIX. The combination of software components is called a database architecture.

TABLE XIX

Primary Goals of DBMS

1. define the chosen database logical structures
2. define the chosen physical structure
3. define user views
4. access the defined database
5. define the storage structures to be used to store the data

---

B. RELATIONAL DATABASE IMPLEMENTATION

Most commercial DBMSs support a single data model [Ref. 48]. The relational model developed during the data analysis stage is implemented using a relational DBMS.

There are three main types of relation DBMSs. There are some based on SQL, others are based on QUEL, and a third category based on other data languages.

One major problem with the implementation of relational DBMSs is the occurence of null values. A null value means

that either the data value is unknown or that the value is inapplicable. Null values present problems when they are values for key columns, when predicates are evaluated, and when relations are joined. It is therefore necessary to identify these potential problem areas and to impose a constraint where null values are prohibited.

C. DATABASE MANAGEMENT SYSTEM SELECTION

It is important at this juncture to divide the problem of DBMS selection into two broad categories. With the recent growth of the microcomputer and the associated growth in application software, a special class of DBMSs has been created. DBMS selection for large systems involves difficult assessment of cost, high risk in terms of the DBMS affecting nearly every aspect of an organization's information processing, and difficulty in determining what kind of data model to use. These same issues are not relevant to the microcomputer user. One significant difference between microcomputer systems and large systems is the size of the database to be managed can be several orders of magnitude larger for large organizations. Another difference is that personal computer systems are intended for use by only a few people. Therefore, all the problems associated with multiple simultaneous access do not arise. Additionally, the general area of backup and recovery, which

is critical to large business systems is given little attention in microcompter applications.

DBMSs for microcomputers, are generally designed for the nonprogrammer user. They differ from DBMSs that are supported by larger computer configurations by the selective power of database access languages and the interface provided to the user. Query languages in personal systems are usually restricted, i.e., each query must address only one file (although many new systems are now capable of multiple file access). The solution is to give users simple commands and require them to formulate complex queries as sequences of such commands. The emphasis of these systems is to provide an interactive interface that enables nonprogrammer users to easily define and populate databases. [Ref. 49]

Most personal computer DBMSs are relatively easy to use. They are designed to provide inexperienced users with an elementary subset of database access commands and then, with experience, to proceed to more sophisticated operations.

dBASE-II is an example of a microcomputer DBMS. It contains three major components to allow database access:

1. a query processor to support on-line ad hoc queries
2. a simple procedural language to store preprogrammed queries
3. a report generator

See Figure 6.2. [Ref. 50]

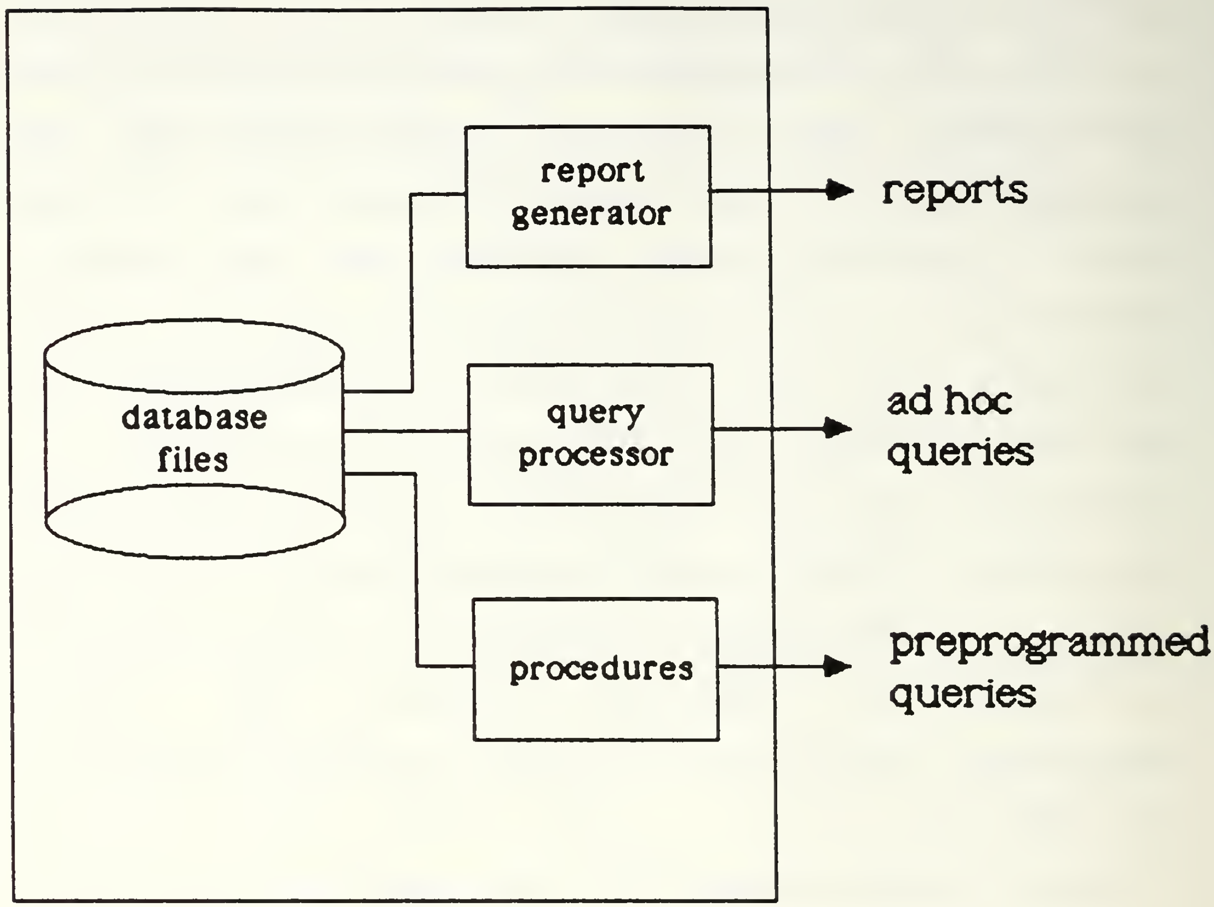


Figure 6.2 Using the Personal Database

Unlike DBMSs for large computer systems. most DBMSs for microcomputers are relational. The dBASE-II query processor supports a SQL-like language. dBASE-II also provides a procedural language to enable preprogrammed queries to be specified. More experienced users can create files that hold such procedures and then execute them as required.

Although DBMSs provided for personal systems usually contain a narrower set of facilites, they are still effective for limited applications.

The most common approach to selecting a DBMS is feature analysis. A list is compiled of general features that characterize DBMSs. A partial listing of example features can be seen in Table XX. Next, each candidate DBMS is evaluated on the basis of each feature. An appropriate ranking system can be applied. The third step involves analyzing all the features in terms of what the organization needs are. The final step involves developing a final score for each of the candidate systems.

The advantages of this method are its simplicity and the appearance of being a rational, quantitative technique. It is also expandable. meaning that additional criteria can be added easily by adding one or more levels of weighting and aggregation between the detailed features and the overall DBMS score. Evaluation can be adjusted to find the optimum set of weights. Disadvantages of this method are that

TABLE XX

Partial Database Management System Feature List

A. VENDOR SUPPORT

1. TRAINING
2. DOCUMENTATION
3. TECHNICAL SUPPORT
4. VENDOR CREDIBILITY
5. USER EXPERIENCE

B. EASE OF USE

1. INITIAL IMPLEMENTATION
2. DATABASE DESIGN CHANGE FACILITIES
3. CONTINUING USE

C. SYSTEM REQUIREMENTS

1. HARDWARE
2. SOFTWARE

D. COMPLETENESS

1. SECURITY FEATURES
2. UTILITIES
3. DATA STRUCTURES SUPPORTED
4. QUERY FACILITIES
5. REPORT WRITER FACILITIES
6. BATCH/ONLINE CAPABILITY
7. COMMUNICATIONS INTERFACE
8. LANGUAGE INTERFACE
9. DATA DICTIONARY CAPABILITY
10. USAGE STATISTICS
11. SUPPORT FOR DATA INDEPENDENCE

E. INTEGRITY

1. CHECKPOINT/RESTART
2. BACKUP/RECOVERY

F. PERFORMANCE

1. CPU USAGE
2. CHANNEL USAGE
3. TUNING FLEXIBILITY

despite its appearance of objectivity. it is still subjective. Both weights and scores are human estimates.

VII. EVALUATING THE DATABASE DESIGN

A. DESIGN OBJECTIVES

Database design must satisfy a certain set of criteria. These criteria can be divided into two major classes: structure and performance criteria.

The first class. structure criteria. concerns the preservation of data properties. Specifically. the preservation of data properties has a high correspondence to normal relations to avoid anomalies. Additionally. the preservation of integrity links between object sets is critical to prevent database inconsistencies.

The second class. performance criteria. concern resource use and database access. Performance criteria include transaction requirements that must be met, a minimum amount of storage should be used. and the number of transfers between memory and storage devices must be minimized.

Due to the nature of this database project. only the first class of criteria will be evaluated. Performance criteria are important in microcomputer DBMSs. however. the ability to control them is often limited. These criteria. on the other hand. are critical to large database system design.

B. INITIAL DESIGN

The goal of initial design is to produce feasible design structure, one that will not be optimized but will satisfy all access requirements.

The design structure must ensure that records needed by on-line transactions can be accessed directly. Otherwise the database system will become bogged down in DML operations to retrieve requested data and the system will be less efficient.

Design methods use a variety of logical and physical design techniques and apply them to realize designs that satisfy the design criteria. These techniques are applied in sequences that depend on the DBMS and on the relative importance placed on the two classes of criteria. [Ref. 51]

C. DESIGN ITERATIONS

Once the initial design is accepted. the designer attempts to improve it. Various design trade-offs are made to reach an optimum design.

After design problems are identified. the designer can select various design tactics to overcome them. Ideally. a set of design tactics are provided for each problem. It is beyond the scope of this paper to go into more detail concerning design iterations and design tactics.

D. EVALUATING DATABASE DESIGN

Database evaluations are made at all the system development steps. Stepwise evlauation, however, will not ensure the optimum final design. Successive steps use the performance estimates to indicate design problems and then design tactics are used to correct these problems. At this point the design is then implemented on the DBMS.

The benefits of database systems are obtained only after the design and implementation are completed, and after sufficient data has been collected so that the user of the database can receive usable information. The remainder of the database cycle will involve assurance of reliability and quality, adaptation to changing requirements, and eventually termination of the operation with transfer of valuable data and procedures to a new system. [Ref. 52]

Once the database design has been finalized and the data loaded, performance improvement involves finding the most efficient route to the data required by each query. This is accomplished by looking at the specific data model and selecting the shortest path. One important question concerns when the access path selection is made. The two choices are: when the retrieval program is written or when it is executed. The other major question is whether the access path decision is made by the DBMS or by either the user or programmer.

One of the basic precepts of the relational model is that access path decisions should be made by the DBMS and. ideally, not until the query is processed. In theory. this should be the optimal approach. Waiting until execution time to make the access decision means that it can take into account the exact state of the database at that moment. Placing responsibility on the DBMS could be viewed merely as a convenience to users, but acually it should be interpreted as an assertion that the system is better able to make these decisions than the user. [Ref. 53] In the microcomputer DBMS. this is especially true considering that most database users fall into the casual user category.

One method of improving database design to increase performance is the addition of extra indices. Normally. an index will be maintained on the values of the key data elements. This allows efficient retrievals where the key values are known because the index can be used to find the relevant records without searching each tuple. It works much the same way as you would use the index in the back of a book when looking for a specific topic. Many systems allow the designer to specify that additional indices should be created and maintained for nonkey data elements that are the basis for frequent retrieval requests.

Indices are not automatically created for all data elements because there are costs involved. They take up storage space in memory. The decision whether to

create additional indices depends on knowing something about the relative frequency of updates and retrievals. If retrievals are far more numerous than updates. then an index should be considered. If updates occur about as often. or more often than retrievals. then the extra work required to maintain the index will probably not be profitable. This illustrates the problems associated with optimizing database design without knowing actual useage requirements.

The bottom line is to make sure that the system supports the user the way it should. If it does not. successive design iterations are performed until it does. See Figure 7.1.

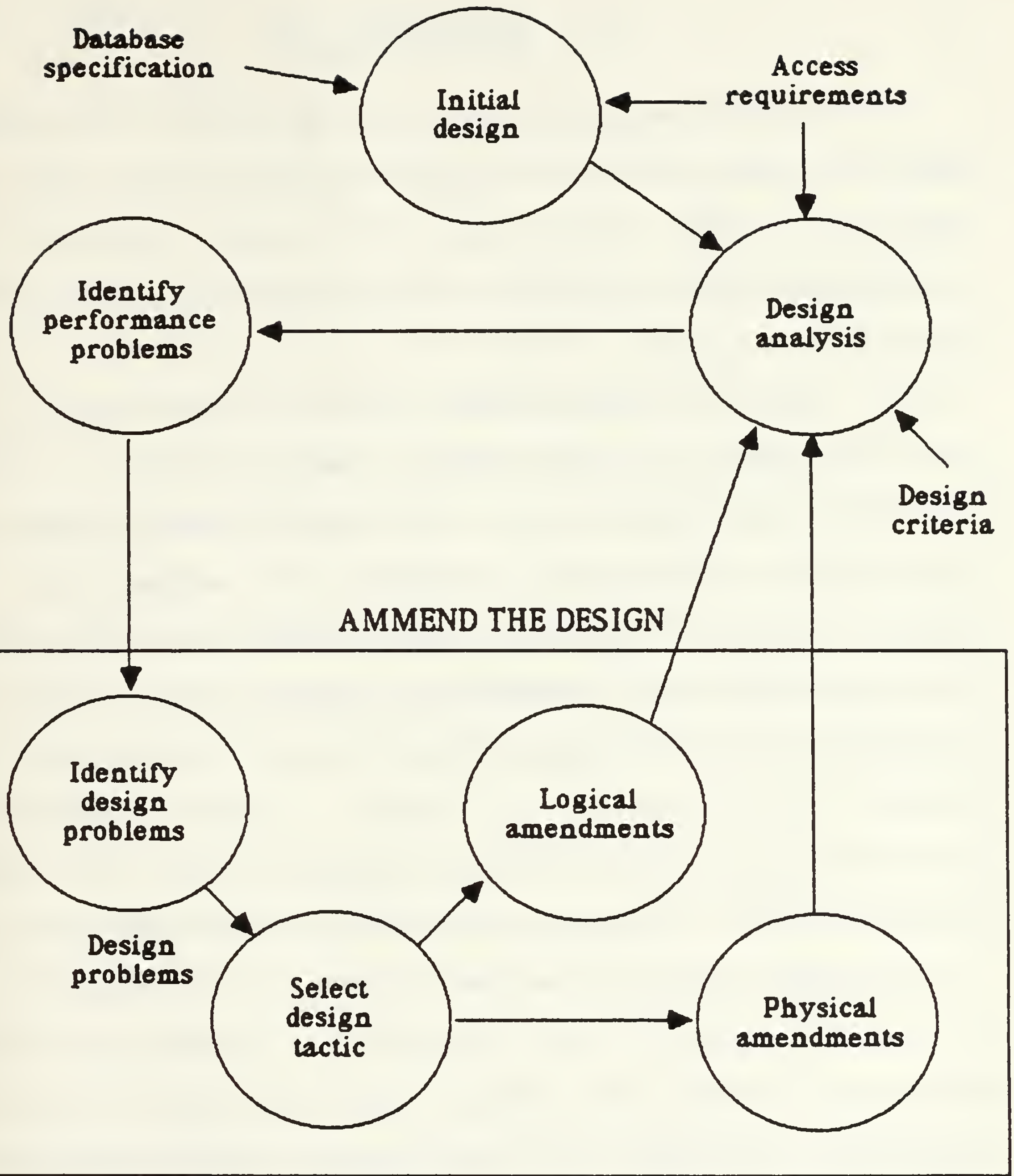


Figure 7.1 Design Iterations

VIII. RECOMMENDATIONS

This thesis proposes an initial design for a relational database system that will support asset identification and valuation as the first step to conduct computer risk assessment at NPS. Much work remains to be done before this system can be used.

The remaining tasks of DBMS selection and implementation would be excellent for student thesis work. I would recommend that the available microcomputer DBMS systems be evaluated and that one be selected for implementing this database design. There are several DBMSs available at NPS including RBASE 4000, POWERBASE, dBASE II and 10BASE. I recommend that a microcomputer DBMS be used because they are designed for users with little computer expertise. Users can also maintain control and security of the system easily (a matter of locking up system and data diskettes).

The DBMSs listed above can handle files of about 6,000 to 10,000 records with "acceptable" delays (10 second retrievals, 1 hour sort times). Additional work is required to determine data quatities to determine the suitability of a microcomputer DBMS given data volumes.

Another important task that must be taken care of is the determination of who will act as database administrator (DBA). The database administrator is

essential for effective operation of the database system. He is responsible for protecting the database while at the same time maximizing benefits to users.

The users of the system must also be identified. It is not clear who will be required to perform the task of data collection and data entry, or who will have to generate reports for the purpose of controlling computer resources.

It is possible, due to the nature of this database, that the user(s) would also act as administrator. With proper database design and careful implementation, and employing a high level language interface, even casual users would be able to manage this system. Naturally, appropriate documentation and training would have to be provided. Also, postdesign optimization and changes to the system would require the skills of a more sophisticated computer technician.

Many of these recommendations hold only if the database system is implemented on a personal computer. If the system is implemented on a larger computer system the problems are much more complex.

One disadvantage of implementing the database system on a personal computer is that access is somewhat restricted. Anyone needing access to the system will have to know where to go and request permission to access the system from the owner. This problem can be alleviated somewhat by creating a computer network and sharing this data between multiple

microcomputers. It is possible to link microcomputers together either with modems and telephone lines or by using commercially available network technology. There are, however, problems with computer networking and database operations. Extensive research is required in this area before an approach could be recommended.

This initial design covers only the asset identification and valuation aspects of computer risk assessment procedures. The global risk assessment database system depicted in Figure 9.1. must also be designed and implemented before the system will provide help with the overall risk assessment process. The initial design has been constructed so that it can be easily expanded to incorporate the global risk assessment process. This decomposition of the database design simplifies the problem by enabling the designer to focus on individual modules one at a time.

IX. CONCLUSIONS

There is growing pressure to control the growth of computers throughout government due to the enormous sums of money that are spent on computer equipment every year. The rapid infiltration of computers into all areas of operations within government, and the Navy in particular, has led to increased reliance on computers to perform its mission. Organizational dependence on computers has made it necessary to continually assess vulnerability to threats related to computer resources.

Numerous directives have been promulgated to help identify threats to computer resources and recommend methodologies to perform risk assessment. The risk assessment procedure, however, requires a great deal of effort and the procedure itself would be well served by employing database technology. A database would greatly enhance the process by providing a standardized, well structured, and maintainable vehicle with which to compute the annual loss expectancy for comuputer resources of a given organization.

A global data structure diagram has been devised to provide such a database. Figures 3.3 and 3.4 show the global system function hierarchy and data structure diagram respectivly. One individual user's view has been designed as

the beginning step toward creating an itegrated database system. The initial design for asset identification and valuation has been presented in this thesis.

The relational model was chosen because its tabular interface can be easily understood by database users and because most microcomputer DBMSs are built on the relational model.

Much work remains to be done on this data model. Users need to be identified and responsibility for the database must be assigned. A DBMS must be selected for implementing the model and data must be collected and entered into the database. These are difficult, time consuming tasks which can be completed by students in the various computer technology curricula.

APPENDIX A

EXAMPLES OF VARIOUS FORMS USED IN THE RISK ASSESSMENT PROCESS

An example of OPNAV 5239/7, ASSET VALUATION WORKSHEET

ASSET VALUATION WORKSHEET

1. ASSET NAME

2. ASSET DESCRIPTION AND JUSTIFICATION OF IMPACT VALUE RATINGS ASSIGNED.

3. IMPACT VALUE RATING BY IMPACT AREA

☐ MODIFICATION ☐ DESTRUCTION ☐ DISCLOSURE ☐ DENIAL OF SERVICE

An example of OPNAV 5239/8. THREAT AND VULNERABILITY EVALUATION WORKSHEET

THREAT AND VULNERABILITY EVALUATION WORKSHEET

1. THREAT NAME

2. DESCRIPTION, EXAMPLES, AND JUSTIFICATION BASED ON EXISTING COUNTERMEASURES AND VULNERABILITES.

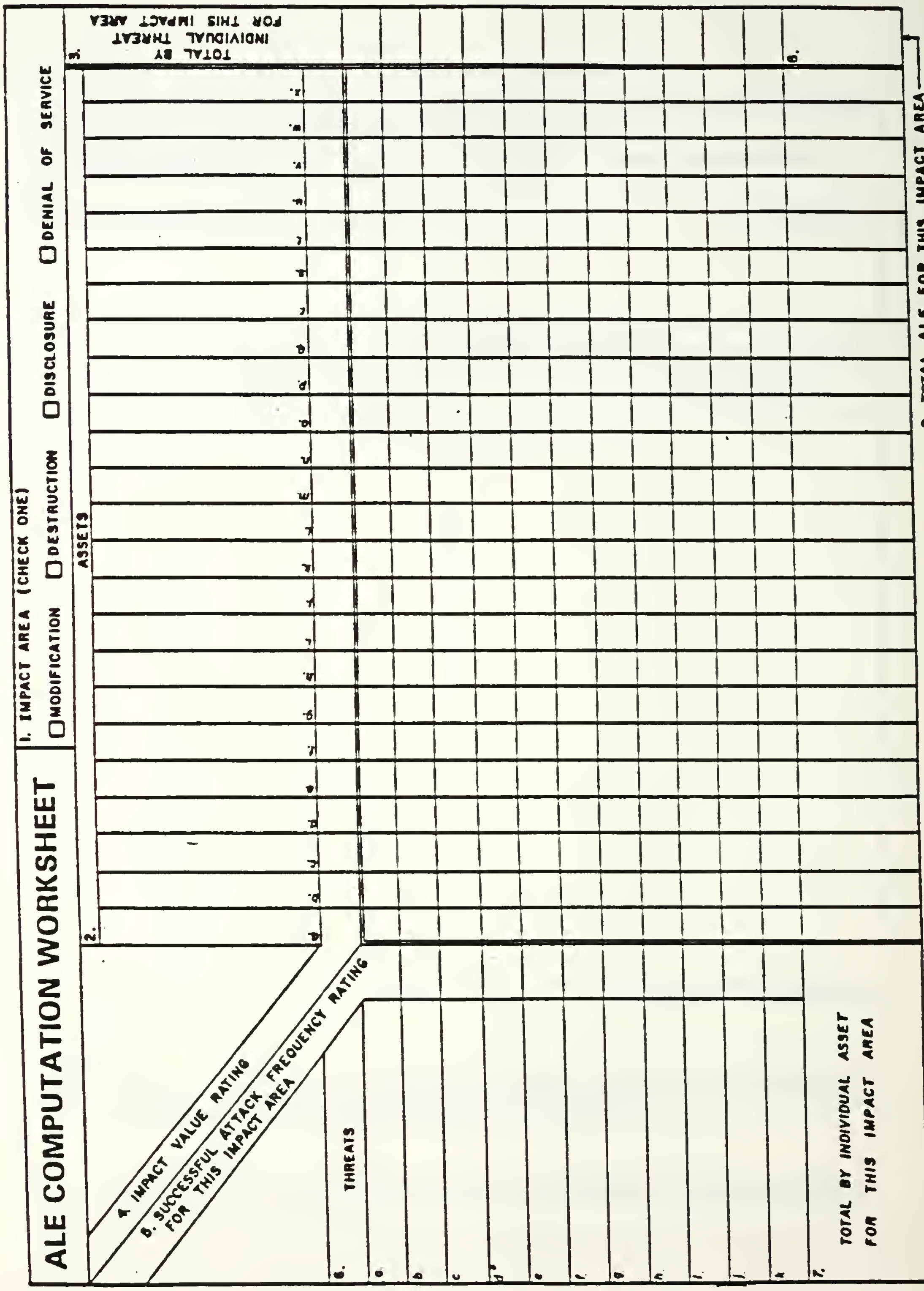
3. SUCCESSFUL ATTACK FREQUENCY RATING BY IMPACT AREA.

☐ MODIFICATION ☐ DESTRUCTION ☐ DISCLOSURE ☐ DENIAL OF SERVICE

An example of OPNAV 5239/9, ALE COMPUTATION WORKSHEET

<table>
<tr><th colspan="4">ADDITIONAL COUNTERMEASURE EVALUATION WORKSHEET</th></tr>
<tr><td colspan="3">1. COUNTERMEASURE NAME</td><td>2. ANNUAL COST</td></tr>
<tr><td colspan="4">3. DESCRIPTION</td></tr>
<tr><td rowspan="2">4. THREATS AFFECTED BY THIS COUNTERMEASURE</td><td colspan="2">5. ALE</td><td rowspan="2">6. ALE SAVINGS</td></tr>
<tr><td>CURRENT</td><td>PROJECTED</td></tr>
<tr><td></td><td></td><td></td><td></td></tr>
<tr><td colspan="3">7. RETURN ON INVESTMENT</td><td>8. TOTAL ALE SAVINGS</td></tr>
<tr><td colspan="4">9. OVERLAPPING ADDITIONAL COUNTERMEASURES</td></tr>
</table>

An example of OPNAV 5239/10, ADDITIONAL COUNTERMEASURE EVALUATION WORKSHEET



An example of OPNAV 5239/11. ADDITIONAL COUNTERMEASURES SUMMARY LISTING

ADDITIONAL COUNTERMEASURES SUMMARY LISTING						
1. ROI		2. ANNUAL COST	3. ALE SAVINGS		4. COUNTERMEASURE	5. MANDATORY REQUIREMENT
A. ORIGINAL	B. ADJUSTED		A. ORIGINAL	B. ADJUSTED		

An example of OPNAV 5239/12, RISK ASSESSMENT MATRIX

RISK ASSESSMENT MATRIX

THREAT	ASSETS AND THEIR IMPACT VALUE							TOTAL ALE BY INDIVIDUAL THREAT
	$	$	$	$	$	$	$	
	TV	TV	TV	TV	TV	TV	TV	
TOTAL ALE BY INDIVIDUAL ASSET								TOTAL ALE

* TV (THREAT VALUE) L (LOW) M (MEDIUM) H (HIGH)

An example of OPNAV 5239/13, ADDITIONAL COUNTERMEASURES SELECTION WORKSHEET

ADDITIONAL COUNTERMEASURES SELECTION WORKSHEET

A. ADDITIONAL COUNTERMEASURES	B. THREATS PAIRED	C. ORIGINAL ALE	D. REVISED ALE	E. ANNUAL SAVINGS	F. ANNUAL COST OF ADDITIONAL COUNTERMEASURES	G. RETURN ON INVESTMENT	H. ADDITIONAL COUNTER-MEASURES PRIORITIES
1.	A.	A.	A.	A.			
	B.	B.	B.	B.			
	C.	C.	C.	C.			
	D.	D.	D.	D.			
	ANNUAL SAVINGS SUBTOTAL			E.	E.	E.	E.
2.	A.	A.	A.	A.			
	B.	B.	B.	B.			
	C.	C.	C.	C.			
	D.	D.	D.	D.			
	ANNUAL SAVINGS SUBTOTAL			E.	E.	E.	E.

APPENDIX B

AN EXAMPLE OF THE ACTIVITY ACCREDITATION SCHEDULE

Activity accreditation schedule sample format

1. NAME & ADDRESS OF ACTIVITY

____

____

____

2. CO'S NAME/TELEPHONE *

____

AUTOVON ____________________

COMMERCIAL ____________________

					8. ACTIVITY ADP ELEMENTS			
5. DAA(S)				6. DATA LEVEL I-II-III	7. MODES OF OPERATION	A. APPLICATION NAME	B. HARDWARE (CPU)MFG	C. SOFTWARE OS	D. FACILITY NAME(S)
CO	NAVDAC	DNI	CNO						

Activity accreditation schedule sample format continued

3. UIC ______________________

4. ADPSO NAME/TELEPHONE *

____

____

AUTOVON ______________________

COMMERCIAL ___________________

8. ACTIVITY ADP ELEMENTS									9. ESTIMATED SCHEDULE				10.
E. COMM. * NODES/ TERMINALS	F. NETWORK NAME(S)	G. TEMPEST REQUIRED		H. COMSEC REQUIRED		I. DES REQUIRED		A. R. A.	B. ST & E	C. CONT PLAN	D. ACCRED-ITATION	NAME OF ADPSSO
		YES	NO	YES	NO	YES	NO					

Activity accreditation schedule sample format

Legend explanation:

1. Name and address of activity

2. Commanding Officer's name and telephone number, AUTOVON and Commercial

3. UIC - Unit Identification Code

4. ADP Security Officer (ADPSO) name and telephone number, AUTOVON and Commercial

Provide the following information (items 5 through 10) for each ADP element of the activity:

5. DAA - Designated Approving Authority - Commanding Officer, COMNAVDAC, Director of Naval Intelligence, or Chief of Naval Operations (OP-942)

6. Level of processed data (I, II, and III)

7. Modes of operation - System high, dedicated, controlled, multilevel

8. ADP element information:

   a. Application name (e.g., payroll, logistics, finance, UADPS Stockpoints, OSIS, SHARE/7, NACMIS, etc.)

   b. Hardware (CPU) manufacturer (e.g., IBM 3081, Univac 1160, etc.)

   c. Software (operating system) (e.g., Univac (Exec 1100))

   d. Facility, building number/room number

e. Communications: Number of nodes (locations). and number of terminals

f. Networks (e.g.. AUTODIN interface. TELNET. ARPANET)

g. TEMPEST required: yes or no, as applicable; if yes, provide TEMPEST task number

h. Is COMSEC required? (yes or no)

i. Is DES required? (yes or no)

9. Estimated schedule for completing accreditation:

a. Risk assessment: estimated start/completion dates

b. ST&E: plan development date; test date

c. Contingency Plan development date

d. Date for submitting request for accreditation

10. Name of ADP systems security officer (ADPSSO)

LIST OF REFERENCES

1. Head. R.. Federal Information Systems Management: Issues and Directions. p. 4. The Brookings Institute. 1981.

2. National Bureau of Standards. Guideline for Computer Security Certification and Accreditation. p. 18. Federal Information Processing Standards Publication 102. September 1983.

3. National Bureau of Standards. Guidelines for Automatic Data Processing Risk Analysis. p. 5. Federal Information Processing Standards Publication 65. August 1979.

4. Ibid.

5. Office of Management and Budget. Security of Federal Automated Information Systems. p. 12. OMB Circular No. A-71. 27 July 1978.

6. Picha. Jill. Budget Analyst. NAVDAC. Washington D. C.. Phone Conversation dated 2 April 1985

7. Office of Management and Budget. Data on Acquisition. Operation. or use of Automatic Data Processing and Telecommunications Systems. Bulletin No. 79-6. January 9.1979.

8. OPNAVINST. 5239.1A dated 3 August 1982.

9. Haase. W. W.. "Data Security Considerations in the Federal Government". p. 18. Seventh Annual Computer Security Conference. November 1980.

10. National Bureau of Standards. Guidelines for Automatic Data Processing Physical Security and Risk Management. Federal Information Processing Standards Publication 31. foreward. June 1974.

11. Office of Management and Budget. Security of Federal Automated Information Systems. p. 11. OMB Circular No. A-71. 27 July 1978.

12. Kroenke. David. Database Processing. p. 3. Science Research Associates. Inc.. 1977.

13. Ibid.. p. 5.

14. Goldstein. Robert. C.. _Database Technology and Management_. p. 7. John Wiley and Sons. 1985.

15. Hawryszkiewycz. I. T.. _Database Analysis and Design_. p. 5, Science Research Associates, 1984.

16. _Ibid_.. p. 15.

17. Sweet. Frank. "What. If Anything. is a Relational Database?". _Datamation_ vol. 30. no. 11. p. 119. July 1984.

18. Hawryszkiewycz. I. T.. _Database Analysis and Design_. p. 36. Science Research Associates. 1984.

19. _Ibid_.. p. 20.

20. Kroenke. David. _Database Processing_. p. 178. Science Research Associates. Inc.. 1977.

21. _Ibid_.. p. 181.

22. _Ibid_.. p. 200.

23. OPNAVINST. 5239.1A dated 3 August 1982.

24. _Ibid_.

25. NAVPGSCOLINST. 5400.2A dated 1 April 1982.

26. Pressman. Roger S.. _Software Engineering: A Practioner's Approach_. p. 107. McGraw-Hill. 1982.

27. Kroenke. David. _Database Processing_. p. 182. Science Research Associates. Inc.. 1977.

28. Hawryszkiewycz. I. T.. _Database Analysis and Design_. p. 94, Science Research Associates. 1984.

29. _Ibid_.. p. 100.

30. _Ibid_.. p. 107.

31. Kroenke. David. _Database Processing_. p. 179. Science Research Associates. Inc.. 1977.

32. Hawryszkiewycz. I. T.. _Database Analysis and Design_. p. 115. Science Research Associates, 1984.

33. Kroenke. David. _Database Processing_. p. 188. Science Research Associates. Inc.. 1977.

34. Ibid.. p. 289.

35. Ibid.. p. 305.

36. Ibid.. p. 309.

37. Ibid.. p. 310.

38. Wiederhold. Gio. Database Design. p. 368. McGraw-Hill, 1977.

39. Ibid.. p. 371.

40. Ibid.. p. 371.

41. Goldstein. Robert. C.. Database Technology and Management. p. 49. John Wiley and Sons. 1985.

42. Kroenke. David. Database Processing. p. 252. Science Research Associates. Inc.. 1977.

43. Hawryszkiewycz. I. T.. Database Analysis and Design. p. 39, Science Research Associates. 1984.

44. Goldstein. Robert. C.. Database Technology and Management. p. 14. John Wiley and Sons. 1985.

45. Ibid.. p. 5.

46. Ibid.. p. 5.

47. Hawryszkiewycz. I. T.. Database Analysis and Design. p. 276. Science Research Associates, 1984.

48. Ibid.. p. 283.

49. Ibid.. p. 326.

50. Ibid.. p. 328.

51. Ibid.. p. 441.

52. Wiederhold. Gio. Database Design. p. 589. McGraw-Hill, 1977.

53. Goldstein. Robert. C.. Database Technology and Management. p. 20. John Wiley and Sons. 1985.

INITIAL DISTRIBUTION LIST

		No. Copies
1.	Library, Code 0142 Naval Postgraduate School Monterey, California 93943	2
2.	CDR. Urbanek Code 44 Naval Postgraduate School Monterey, California 93943	1
3.	Prof. N. Lyons Code 54LB Naval Postgraduate School Monterey, California 93943	1
4.	LCDR. P.S. Fischbeck Code 55FB Naval Postgraduate School Monterey, California 93943	1
5.	Naval Postgraduate School Computer Technology Curricular Office Code 37 Monterey, California 93943	1
6.	Defense Technical Information Center Cameron Station Alexandria, Virginia 22314	2
7.	LCDR Fred W. Thompson, Jr. 3 North Parkway Elkton, MD 21921	2

213162

Thesis
T432 Thompson
c.1

Initial design of a relational database system for NPS computer asset identification and valuation for Risk Assessment.